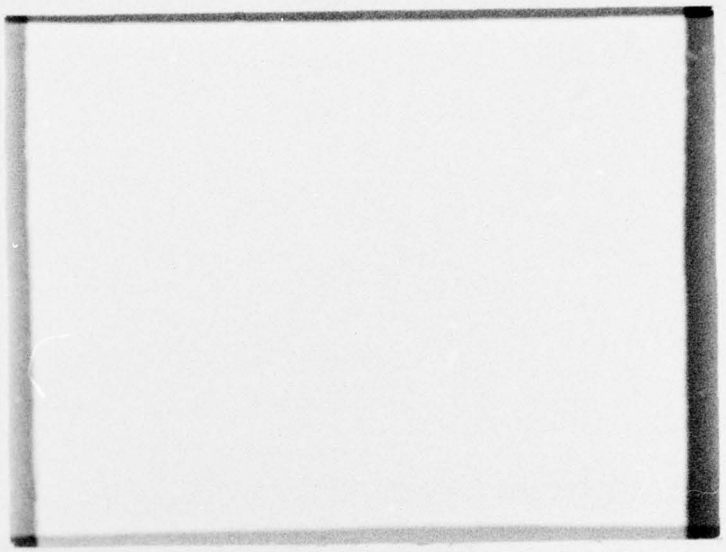
AD-A045 117 CONSAD RESEARCH CORP PITTSBURGH PA F/G 5/9
AN OVERVIEW OF THE PROTOTYPE INTEGRATED SIMULATION EVALUATION M--ETC(U)
APR 77 C R EISELE, C D LAIDLAW F44620-76-C-0125
UNCLASSIFIED AFOSR-TR-77-1006 NL

1 OF 1
AD A045117

END
DATE FILMED
11-77
DDC

CONSAD Research Corporation



AIR FORCE OFFICE OF SCIENTIFIC RESEARCH (AFSC)
NOTICE OF TRANSMITTAL TO DDC
This technical report has been reviewed and is approved for public release IAW AFR 190-12 (7b).
Distribution is unlimited.
A. D. BLOSE
Technical Information Officer

(18) (19) AFOSR-TR-77-1006

(9) FINAL REPORT.

(6) An Overview of the Prototype Integrated Simulation Evaluation Model of the Air Force Manpower and Personnel System.

APPENDIX A. A SOURCE LISTING OF THE PROGRAM CODE FOR ISEM-P.

Contract Number (15) F44620-76-C-0125

(10) Charles R. Eisele
Charles D. Laidlaw

(16) 2313

Prepared for: (17) A3

Directorate of Life Sciences
Air Force Office of Scientific Research
Attention: NL
Building 410
Bolling Air Force Base, D.C. 20332

Prepared by:

CONSAD Research Corporation
121 North Highland Avenue
Pittsburgh, Pennsylvania 15206

(11) 27 April 1977

(12) 90p.

Approved for public release; distribution unlimited.

1473

387 958

GB

APPENDIX A: Source Listing of Program Code for ISEM-P

```
PREAMBLE**  FOR PROTOTYPE ISEM MODEL
 NORMALLY MODE IS INTEGER
**
**
**DEFINES TO ASSURE UNIQUENESS OF NAMES TO 5 OR 6 CHARS
DEFINE CHECK.PIPELINE TO MEAN CH.TP
DEFINE CHECK.PIPE TO MEAN CH.TP
DEFINE CHECK.TRAVELPIPE TO MEAN CH.TP
DEFINE AIRMEN.NAME TO MEAN AN.NM
DEFINE AIRMEN RECRUITS TO MEAN AR.RT
DEFINE AIR.INITIAL.YG.PCT TO MEAN AI.YP
DEFINE AIR.OFF.EQUIV TO MEAN AO.EQ
DEFINE ASSIGN.CHECK TO MEAN AS.CK
DEFINE ASSIGN.OJT TO MEAN AS.OJ
DEFINE ASSIGN OUT TO MEAN AS.OU
DEFINE ASSIGN SCH TO MEAN AS.SC
DEFINE ASSIGN.SEP TO MEAN AS.SP
DEFINE BASE.LOC TO MEAN BS.LC
DEFINE BASE.PRIORITY.SET TO MEAN BS.PS
DEFINE BASE.SUPPORT.STAFF TO MEAN BS.SS
DEFINE BASE.TYPE TO MEAN BS.TP
DEFINE FLOW.IN TO MEAN FL.IN
DEFINE FLOW.OUT TO MEAN FL.OU
DEFINE FLOW.SEP TO MEAN FL.SP
DEFINE INIT.PERIODS TO MEAN IN.PD
DEFINE INIT.PERIOD0 TO MEAN IN.P0
DEFINE INIT.YEAR0 TO MEAN IN.Y0
DEFINE LEVEL.ENTER TO MEAN LV.EN
DEFINE LEVEL.TAUGHT TO MEAN LV.TT
DEFINE MAKE.ARLK TO MEAN MK.AR
DEFINE MANPOWER.PE TO MEAN MP.PE
DEFINE MISSION.SKILLS TO MEAN MS.SK
DEFINE MISSION.SUPPORT.SKILLS TO MEAN MS.SS
DEFINE OFF.INITIAL.YG.PCT TO MEAN OF.YP
DEFINE OFFICER.NAME TO MEAN OF.NM
DEFINE OFFICER.RECRUITS TO MEAN OF.RC
DEFINE OFFICER.SKILL1 TO MEAN OF.S1
DEFINE OJT.FLOW.IN TO MEAN OJ.FI
DEFINE OJT.FLOW.OUT TO MEAN OJ.FO
DEFINE OJT.OUT TO MEAN OJ.OU
DEFINE OJT.PCT TO MEAN OJ.PC
DEFINE OUTPUTS TO MEAN OS.OS
DEFINE PERIOD0 TO MEAN PD.P0
DEFINE PIPE.EXIT TO MEAN PP.EX
DEFINE PLAN.SAVING TO MEAN PL.SV
DEFINE RETENTION.RATE TO MEAN RT.RT
DEFINE RETENTION.VARIANCE TO MEAN RT.VR
DEFINE ROTATION.CYCLE TO MEAN RO.CY
DEFINE ROTATION.MEMORY TO MEAN RO.MM
DEFINE ROTATION.POOL TO MEAN RO.PL
DEFINE SCHOOL.CAPACITY TO MEAN SC.CA
DEFINE SCHOOL.CHART TO MEAN SC.CH
DEFINE SCHOOL.QUEUE.SIZE TO MEAN SC.QS
DEFINE SCHOOL.QUEUE TO MEAN SCH.Q
```

```
DEFINE SKILL.ADJUSTMENT TO MEAN SK.AD
DEFINE SKILL.NAME TO MEAN SK.NM
DEFINE SKILL.TAUGHT TO MEAN SK.TT
DEFINE TECH.TRAINING TO MEAN TH.TT
DEFINE TOTAL.GRADUATION TO MEAN TO.GR
DEFINE TOTAL.NEW.STUDENTS TO MEAN TO.NS
DEFINE TOTAL.SCHOOL.QUEUE TO MEAN TO.SQ
DEFINE TOTAL.WAIT.QUEUE TO MEAN TO.WQ
DEFINE TRAINING.CONSTANT TO MEAN TR.CT
DEFINE TRAINING.PROJECTION TO MEAN TR.PR
DEFINE TRAINING.SCHEDULER TO MEAN TR.SC
DEFINE TRAINING.TIME TO MEAN TR.TM
DEFINE TRAVEL.VOLUME TO MEAN TR.VL
DEFINE USAF.MANPOWER.DESIRED TO MEAN US.MD
DEFINE USAF.MIN.MANPOWER TO MEAN US.MM
DEFINE USAF.MISSION.AUTH TO MEAN US.MA
DEFINE USAF.PCHANGE TO MEAN US.PC
DEFINE USAF.PROJECTION TO MEAN US.PR
DEFINE USAF.SCHOOL.AUTH TO MEAN US.SA
DEFINE USAF.UPGRADES TO MEAN US.UP
DEFINE YEAR.PLAN TO MEAN YR.PL
''
''GLOBAL PROPERTIES OF AIRFORCE FACILITIES.....................
''
PERMANENT ENTITIES
  EVERY BASE HAS
     A NAME,                    ''ALPHA NAME FOR REPORTS
     A CONT.LOCATION(*/15),          ''CONUS, EUR, OR PAC
     A SEP.POINT(*/12),         ''BASE OF DEBARKATION FOR SEPARATEES
     AN OVERSEAS(*/15),     ''FLAG INDICATING OVERSEAS LOCATION.
     A BASE.TYPE(*/10),      ''OPS,TT,BMT,UPT.
     A STATE.LOCATION,
     A MAJCOM
     AND BELONGS TO A BASE.PRIORITY.SET
TEMPORARY ENTITIES.........
  EVERY AIRFORCE
     HAS SOME OFFICERS      AND SOME AIRMEN
     AND OWNS A BASE.PRIORITY.SET
DEFINE ALL.THE.AIRMEN AND ALL.THE.OFFICERS AS 1-DIM INTEGER ARRAYS
DEFINE BASE.PRIORITY.SET AS A SET RANKED BY LOW BASE.TYPE
''
PERMANENT ENTITIES
  EVERY SCHOOL HAS
     A SKILL.TAUGHT(*/12),    ''WHAT SKILL THIS SCHOOL PROVIDES
     A LEVEL.TAUGHT(*/12),    ''AT WHAT LEVEL
     A BASE.LOC(*/12),        ''WHAT BASE THIS SCHOOL RESIDES AT
     A DURATION,              ''DURATION OF TRAINING COURSE
     A SCHOOL CAPACITY(*/5)         ''MAX POSSIBLE ENROLLMENT
  DEFINE DURATION AS A REAL VARIABLE
```

```
**
**
**THE FACILITIES SUPPORT VARIOUS MISSIONS (E.G. AIRCRAFT SQUADRONS)
**EACH TYPE OF MISSION REQUIRES MANPOWER HAVING VARIOUS SKILLS
**  AT VARIOUS LEVELS.  THE AMOUNT OF MANPOWER DEPENDS ON THE
**  AMOUNT OF EQUIPMENT, HOW MUCH TIME IT IS IN USE, ETC.
**
PERMANENT ENTITIES
  EVERY MISSION HAS
     SOME STD.EQUIPMENT,
     SOME STD.FLYING.HRS,
     A STD.MUNITIONS,
     A FLT.PLAN.COEFF,
     AN MH.PER.EH,
     A MUNITIONS.LF,
     SOME SKILL.COEFF
  DEFINE AIR.OFF.EQUIV AS A 1-DIM INTEGER ARRAY  **POINTS TO AIR SK
  DEFINE MONTHLY.MAN.HRS AS AN INTEGER VARIABLE
  DEFINE MANPOWER.PF AS A REAL VARIABLE
  DEFINE FLT.PLAN.COEFF , STD.MUNITIONS AS REAL VARIABLE
  DEFINE UPT.CAP.FACTOR,UNT.CAP.FACTOR,TT.CAP.FACTOR,
         BMT.CAP.FACTOR,OTS.CAP.FACTOR AS REAL VARIABLES
  DEFINE IP.SHARE,IN.SHARE AS 1-DIM REAL ARRAYS
  DEFINE INT.ROT.PCT TO MEAN INITIAL.ROTATION.PCT
**
  EVERY SKILL AND LEN.TEXT HAS
     A SKILL.NAME        **ALPHA NAME FOR REPORTS
**
  EVERY LEVEL HAS
     AN OFFICER.NAME,    **ALPHA NAME FOR REPORTS
     AN AIRMEN.NAME      **ALPHA NAME FOR REPORTS
**
**
**MISSIONS ARE ASSIGNED TO BASES AS "OUTPUTS" OF THAT BASE.
**THE ASSIGNMENTS MAY VARY FROM YEAR TO YEAR.
**
  EVERY YEAR AND BASE OWNS
     SOME OUTPUTS,       **LIST OF ASSIGNED OUTPUTS
     AND HAS
     A YE.MANPOWER.DESIRED(*/3), **SUM OVER ALL OUTPUTS (S,L)
     A YE.MIN.MANPOWER(*/3)      **SUM OVER ALL OUTPUTS (S,L)
**
TEMPORARY ENTITIES
**
  EVERY OUTPUT HAS
     A TYPE,              **TYPE OF MISSION
     A QUANTITY,          **NUMBER OF THESE MISSIONS ASSIGNED TO BASE
     A UTILIZATION,       **USUALLY, # OF FLYING HOURS/AIRCRAFT/MO
     AND BELONGS TO
     AN OUTPUTS
**
```

```
''
''
''AGGREGATE-PLANNING SUBMODEL.....
''(1) DEVELOPS TOTAL USAF AND INDIVIDUAL BASE AUTHORIZATIONS
''   FROM CONGRESSIONAL AUTHORIZATIONS AND GIVEN MANPOWER POMTS.
''(2)CONSTRUCTS YEARLY PLANS FOR TRAINING AND RECRUITMENT TO MAKE
''   SUPPLY OF MANPOWER MEET AUTHORIZATION, COMPENSATING FOR
''   SEPARATIONS THAT OCCUR DURING THE YEAR.
''
PERMANENT ENTITIES
''
  EVERY YEAR HAS
     SOME YEAR.GROUPS,    ''DISTRIBUTION OF MEN BY (S,L,YR.GROUPS))
     A YES.AIRMEN,        ''CONGRESSIONS AUTHORIZATION FOR AIRMEN
     A YES.OFFICERS    ''     "                "     FOR OFFICERS
  DEFINE AIR.AUTH.TRAJECTORY,OFF.AUTH.TRAJECTORY,
         AIR.INIT.AUTH.PCT,OFF.INIT.AUTH.PCT AS REAL VARIABLES
''
  EVERY YEAR AND BASE HAS
     A YE.AUTHORIZATION(*/3)
''
  NORMALLY MODE IS REAL
''
  EVERY YEAR, SKILL, AND LEVEL HAS
    . AN EXPECTED.SEPARATIONS,''TOTAL SEPS (S,L)
     AN ACTUAL.SEPARATIONS,''TRUE SEPS (S.L)
     A USAF.MISSION.AUTH,
     A CONUS.USAF.M.AUTH,
     A USAF.SCHOOL.AUTH,
     A USAF.MIN.MANPOWER,
     A USAF.PROJECTION,
     A DESIRED.LSPLIT,
     A USAF.MANPOWER.DESIRED,
     A USAF.UPGRADES,
     A USAF.PCHANGE
''
  EVERY SKILL AND LEVEL HAS
     A RETENTION.VARIANCE
  DEFINE RETENTION.VARIANCE AS A REAL ARRAY
''
  EVERY SKILL HAS A TRAINING.CONSTANT
''
''
  DEFINE RETENTION.RATE AS A 3-DIM REAL ARRAY
  DEFINE YG.SPLIT   ''DISTRIBUTION OF CROSS-LVL CHANGES BY YG
     AS A 3-DIM REAL ARRAY
  DEFINE YEARD AS AN INTEGER VARIABLE
  DEFINE MAX.LAG AS AN INTEGER VARIABLE
  NORMALLY DIMENSION IS 3, MODE IS INTEGER
  THE SYSTEM HAS
     A YG.ARRAY(*/5),     ''USED TO RESERVE INITIAL YR GRP MEM
     A THISYR(*/5),       ''USED IN CALCULATING SEPS FROM YG MEM
     A NEXTYR(*/5)        ''  "       "        "      "
  DEFINE YG.ARRAY,THISYR,NEXTYR AS INTEGER VARIABLES
  NORMALLY DIMENSION IS 0
```

```
''
''
''
''
''TRAINING SCHEDULER
''GIVEN A SEQUENCE OF YEAR PLANS, AND THE MONTHLY DISTRIBUTION
''OF SEPARATIONS, WE
''(1) DEVELOP A SCHEDULE OF SLOTS TO BE FILLED ("HOLES")
''      BY MONTH, SKILL, AND LEVEL FOR THE ENTIRE AIRFORCE.
''(2)CALCULATE THE NUMBER OF AIRMEN AND OFFICER RECRUITS THAT
''      MUST ENTER THE AIRFORCE EACH MONTH TO FILL THE SLOTS
''      CREATED AT THE LOWEST LEVELS
''
  DEFINE EXIT.F AS A 1-DIM REAL ARRAY
  DEFINE USAF.HOLES AS A 3-DIM REAL ARRAY
  DEFINE AIRMEN.RECRUITS AND OFFICER.RECRUITS AS 1-DIM INTEGER
     ARRAYS
  DEFINE INITIAL.ROTATION.PCT AS A 2-DIM REAL ARRAY

  EVERY SKILL AND LEVEL HAS
     AN OJT.PCT,
     A TRAINING.TIME
  DEFINE OJT.PCT, TRAINING.TIME AS REAL VARIABLES
''
''
''OUTPUT REQUIREMENTS FOR AGGREGATE PLANNING SUBMODEL ..................
''
''
''
''
''
''
''ASSIGNMENT-PLANNING SUBMODEL.....................................
''
''(1) TRANSLATES USAF YE AUTHORIZATIONS INTO BASE AUTHORIZATIONS
''        BY MONTH.
''(2) TRANSLATES YEAR PLAN INTO MONTHLY TARGETS FOR TOTAL USAF:
''      >UPGRADES, SPLIT INTO OJT AND TECH TRAINING RQMTS
''      >SEPARATIONS EXPECTED
''(3) DEVELOPS PROJECTIONS OF BASE MANPOWER LEVELS BY MONTH
''(4) DEVELOPS PROJECTIONS OF SCHOOL OUTPUTS BY MONTH
''(5) PRODUCES THE TRAVEL ORDERS THAT WILL:
''      >TRANSFER SCHOOL OUTPUTS TO BASES (OR OTHER SCHOOLS)
''      >SATISFY OVERSEAS ROTATION RQMTS
''      >TRANSFER BASE PERSONNEL TO SCHOOL FOR UPGRD TRAINING
''
PERMANENT ENTITIES
''
  NORMALLY MODE IS INTEGER
''
  EVERY PERIOD AND BASE HAS
     AN AUTH.SUPPLY(*/3),     ''BASE'S AUTHORIZATION
     A MIN.SUPPLY(*/3)        ''MINIMUM NEEDED MANPOWER TO FUNCTION
''
```

```
  EVERY PERIOD AND SCHOOL HAS
     A TECH.TRAINING.POOL(*/6), ''PROJECTED SCHOOL OUTFLOW THIS PERIOD
     AN ENTER SCHOOL.TABLE(*/6)  ''PROJECTED SCHOOL INFLOW THIS PERIOD
''
  EVERY PERIOD AND BASE HAS
     A PROJECTION(*/3),      ''PROJECTED SUPPLY THIS MONTH
     A ROTATION.POOL(*/3)    '' # STILL TO BE ROTATED OUT BY THIS PERIOD
     AND OWNS
     SOME PLANNED.ASSIGNMENTS''PLANNED TRAVEL OUT OF  BASE
                              ''THIS PERIOD
''
  EVERY SCHOOL OWNS
     A DISPOSITION      ''SET OF TRAVEL ORDERS TO BE USED BY GRADS
''
  NORMALLY DIMENSION IS 3
  THE SYSTEM HAS A DEMAND (*/5)
  NORMALLY DIMENSION IS 0
''
TEMPORARY ENTITIES

  EVERY ABLK ''IS A TRAVEL ORDER FOR A BLOCK OF MEN AND
     HAS
     A SKL (1/6) IN WORD 1,
     A CLASS (2/6) IN WORD 1,
     A LVL (5/12) IN WORD 1,
     A PURPOSE (6/12) IN WORD 1,
     A DESTINATION (4/6) IN WORD 1,
     A SIZE (4/4) IN WORD 1,
     A DEPARTURE.DATE IN WORD 2,
     AND MAY BELONG TO
        A DISPOSITION,
        SOME PLANNED.ASSIGNMENTS
  DEFINE DEPARTURE.DATE AS A REAL VARIABLE
''
EVENT NOTICES INCLUDE ASSIGNMENT
''
  DEFINE SCHOOL.CHART  ''RELATES SKILLS AND LEVELS TO SCHOOLS
     AS A 2-DIM INTEGER ARRAY
  DEFINE N.AIRMEN.SKILLS, OTS, BMT AS INTEGER VARIABLES  .
''
  DEFINE PERIOD1,      ''LAST PERIOD OF YEAR0
         N.TECH.SCHOOLS,''N.SCHOOL LESS OTS AND BMT
         OFFICER.SKILL1,''FIRST OFFICER SKILL #
         H.MONTHS      ''# OF MONTHS FROM TIME.V TO HORIZON
         AS INTEGER VARIABLES
  DEFINE OTS.TIME, BMT.TIME     ''BASIC TRAINING TIMES
         AS REAL VARIABLES
  DEFINE OJT.DELAY     ''OJT ELIGIBILITY LAGS
         AS A 1-DIM REAL ARRAY
''
  DEFINE OJT.OUT, SEP.OUT, TT.OUT AS INTEGER FUNCTIONS
''
```

```
''
''OUTPUT REQUIREMENTS FOR ASSIGNMENT PLANNING SUBMODEL ..................
''
''
PERMANENT ENTITIES
''
  DEFINE : TO MEAN IN WORD
  EVERY BASE,SKILL,LEVEL HAS
     AN ASSIGN.OUT(1/5) : 1,          ''SCHEDULED ASSIGNMENT DEPARTURES
     AN ASSIGN.OJT(2/5) : 1,          ''SCHEDULED UPGRADES BY OJT
     AN ASSIGN.SEP(3/5) : 1,          ''SCHEDULED SEPERATION DEPARTURES
     AN ASSIGN.SCH(4/5) : 1           ''SCHEDULED DEPARTURES FOR SCHOOL
''
  EVERY SCHOOL HAS A LEVEL.ENTER(*/12)
''
EVENT NOTICES INCLUDE SAVE.ASSIGN
''
BEFORE FILING IN PLANNED.ASSIGNMENTS, CALL ASSIGN.CHECK
''
''
''
''
''
''PERSONNEL-FLOW SUBMODEL.....................................
''
''IMPLEMENTS THE PLAN DEVELOPED ABOVE.
''(1)  CAUSES SEPARATIONS TO OCCUR
''(2)  CAUSES RECRUITS TO BE INDUCTED
''(3)  CAUSES TRAVEL TO OCCUR
''(4)  CAUSES GRADUATIONS FROM SCHOOLS TO OCCUR
''THE EFFECT OF THESE EVENTS IS TO MOVE BLOCKS OF MEN INTO AND
''OUT OF BASE SUPPLIES (OR SCHOOL ENROLLMENTS) VIA TRAVEL PIPELINES
''
''
PERMANENT ENTITIES
''
  EVERY BASE HAS
     A SUPPLY(*/3),           ''CURRENT BASE POPULATION BY (S,L)
     AN OJT.ELIGIBLES(*/3),       ''CURRENT # AVAILABLE FOR OJT UPGRD
     AN OJT.INELIGIBLES(*/3), ''RECENT UPGRADES(*/3), NOT READY FOR OJT
     A  PENDING.OJT,       ''TOTAL PLANNED OJT,, TIME.V TO HORIZON
     A ROTATION.MEMORY(*/3), ''3-D ARRAY OF ROATATION HISTORY FOR OVERSE
     A ROTATION.CYCLE(*/3)    ''DURATION OF ROTATION CYCLE IN MONTHS
''
  EVERY SCHOOL HAS
    AN ENROLLMENT(*/5),     ''CURRENT NUMBER OF STUDENTS IN PROGRESS
    A SCHOOL.QUEUE.SIZE(*/5)
     AND OWNS
    A SCHOOL.QUEUE       ''LIST OF STUDENTS WAITING TO ENROLL
''
```

```
  EVERY TRAVELPIPE HAS
     A CAPACITY(*/4),   ''MAX # OF PERSONS SIMULTANEOUSLY IN PIPE
     A VOLUME(*/4),     ''CURRENT    "      "        "          " "
     A MIN.TRAVEL.TIME,''MIN TIME A TRAVELER SPENDS IN PIPE
     AN EXIT.BASE(*/12),''WHERE THE PIPE EMPTIES ITS FLOW
     A WAIT.QUEUE.SIZE(*/4),
     AND OWNS
     A WAITING.QUEUE    ''PERSONS WAITING TO TRAVEL
  DEFINE MIN.TRAVEL.TIME AS A REAL VARIABLE
''
  DEFINE PIPE.CHART  ''RELATES PAIRS OF BASES TO THE PIPE BETWEEN
     AS A 2-DIM INTEGER ARRAY      ''THEM
''
TEMPORARY ENTITIES
  EVERY ABLK MAY BELONG TO
     A SCHOOL.QUEUE,
     A WAITING.QUEUE
  AND HAS
     A (P.DISPOSITION(1/3),S.DISPOSITION(2/3),
        M.DISPOSITION(3/3)) IN WORD 3,
     A (P.PLANNED.ASSIGNMENTS(1/3),
        S.PLANNED.ASSIGNMENTS(2/3),
        M.PLANNED.ASSIGNMENTS(3/3)) IN WORD 3,
     A (P.WAITING.QUEUE(1/3),
        S.WAITING.QUEUE(2/3),
        M.WAITING.QUEUE(3/3)) IN WORD 3,
     A (P.SCH.Q(1/3),
        S.SCH.Q(2/3) ,
        M.SCH.Q(3/3)) IN WORD 3
''
EVENT NOTICES INCLUDE
  INDUCTION AND MOVEMENTS
''
  EVERY GRADUATION HAS
     A G.BLK,
     A G.SCH
''
  EVERY PIPE.EXIT HAS
     A P.BLK,
     A PIPE
''
 DEFINE PLAN.YEAR AS AN INTEGER FUNCTION WITH 1 ARGUMENT
 DEFINE GET.GRADUATES AS AN INTEGER FUNCTION WITH 4 ARGUMENTS
 DEFINE GET.ROTATIONS AS AN INTEGER FUNCTION WITH 4 ARGUMENTS
 DEFINE PERIOD.F AS AN INTEGER FUNCTION WITH 1 ARGUMENT
 DEFINE SKILL.NAME, OFFICER.NAME, AIRMEN.NAME AS ALPHA VARIABLES
 DEFINE NAME,STATE.LOCATION,MAJCOM AS ALPHA VARIABLES
DEFINE YD.FRAC,MIN.MANPOWER.FRAC,OFF.INITIAL.YG.PCT,
       AIR.INITIAL.YG.PCT AS 2-DIM REAL ARRAYS
''
''
```

```
**OUTPUT REQUIREMENTS FOR THE PERSONNEL FLOW SUBMODEL ..................
**
**
BEFORE REMOVING FROM PLANNED.ASSIGNMENTS, CALL FLOW1
AFTER SCHEDULING A GRADUATION, CALL FLOW2
BEFORE DESTROYING AN ABLK, CALL FLOW3
BEFORE FILING IN SCHOOL.QUEUE, CALL FLOW4
AFTER SCHEDULING A PIPE.EXIT, CALL FLOW5
BEFORE FILING IN WAITING.QUEUE, CALL FLOW6
**
EVENT NOTICES INCLUDE PERSONNEL.SAVING,STORAGE.MANAGER,MASSIVE.RIF
**
PERMANENT ENTITIES
**
  EVERY BASE,SKILL,LEVEL HAS
     A FLOW.SEP(5/5) : 1,          **SEPERATION DEPARTURES
     AN OJT.FLOW.OUT(1/5) : 2,     **LEVEL UPGRADES BY OJT
     A TO.SCH.FLOW(2/5) : 2,       **DEPARTURES TO SCHOOL
     A FLOW.OUT(3/5) : 2,          **OTHER ASSIGNMENT DEPARTURES
     A FLOW.IN(4/5) : 2,           **OTHER ASSIGNMENT ARRIVALS
     AN OJT.FLOW.IN(5/5) : 2       **UPGRADES INTO SKILL LEVEL SUPPLY BY OJ
**
  EVERY SCHOOL HAS
     A TOTAL.SCH.QUEUE,        **SIZE OF WAITING QUEUE FOR SCHOOL
     A PEAK.SCH.QUEUE,         **LARGEST VALUE OF WAITING QUEUE SIZE
     A TOTAL.NEW.STUDENTS,     **NUMBER ENTERING THE SCHOOL
     A TOTAL.GRADUATION,       **NUMBER LEAVING THE SCHOOL
     A PEAK.ENROLLMENT         **LARGEST SCHOOL ENROLLMENT
**
  EVERY TRAVELPIPE HAS
     A TRAVEL.VOLUME,          **NUMBER IN TRANSIT
     A PEAK.VOLUME,            **LARGEST NUMBER IN TRANSIT
     A TOTAL.WAIT.QUEUE,       **NUMBER AWAITING DEPARTURE FOR TRAVEL
     A PEAK.WAIT.QUEUE         **LARGEST NUMBER AWAITING DEPARTURE
**
**
  **SIMULTANEOUS EVENTS IN THIS ORDER
  PRIORITY ORDER IS PERSONNEL.SAVING,   **FOR PREVIOUS MONTH
                    STORAGE.MANAGER,
                    PIPE.EXIT,
                    GRADUATION,

                    ASSIGNMENT,
                    SAVE.ASSIGN,
                    MOVEMENTS
**
**
**GENERAL  UTILITY ROUTINE, GLOBAL VARIABLES, AND CONSTANTS
**
  DEFINE CEIL.F, CALENDAR.MONTH, LEN.F,CELL.F AS INTEGER FUNCTIONS
  DEFINE TIME.F AS A REAL FUNCTION
  DEFINE YRS.TO.SIMULATE  **LENGTH OF SIMULATION RUN IN YEARS
         AS AN INTEGER VARIABLE
  DEFINE N.MONTHS AS AN INTEGER VARIABLE
```

```
**SYMBOLIC CONSTANTS:
 DEFINE LEVEL1 TO MEAN 1
 DEFINE LEVEL2 TO MEAN 2
 DEFINE LEVEL3 TO MEAN 3
 DEFINE LEVEL4 TO MEAN 4
 DEFINE LEVEL5 TO MEAN 5
  DEFINE OPS TO MEAN 4
  DEFINE TTRB,UPTB,BASICB,APOE.EAST,APOE.WEST
         AS INTEGER VARIABLES
  DEFINE PILOTF TO MEAN 52 **FIRST PILOT SKILL
  DEFINE PILOTL TO MEAN 58 **LAST PILOT SKILL
  DEFINE NAVF TO MEAN 59  **FIRST NAVIGATOR SKILL
  DEFINE NAVL TO MEAN 66  **LAST NAVIGATOR SKILL
  DEFINE PACAF TO MEAN 3
  DEFINE USAFE TO MEAN 2
  DEFINE MAC TO MEAN 1
  DEFINE TAC TO MEAN 4
  DEFINE SAC TO MEAN 5
  DEFINE CONUS TO MEAN 0

 DEFINE YG1 TO MEAN 1
 DEFINE YG30 TO MEAN 30
 DEFINE N.YEAR.GROUP TO MEAN 30
  DEFINE N.YEAR.GROUPS TO MEAN 30

DEFINE SKILL.CONSTANT TO MEAN SKILL.COEFF
DEFINE SKILL.CONSTANTS TO MEAN SKILL.COEFF


  DEFINE N.SKILL.TYPES TO MEAN 91

 DEFINE TRAINEE TO MEAN 0
 DEFINE HELPER TO MEAN 1
 DEFINE APPRENTICE TO MEAN 2
 DEFINE JOURNEYMAN TO MEAN 3
 DEFINE TECHNICIAN TO MEAN 4
 DEFINE SUPERVISOR TO MEAN 5

 DEFINE LEVYING TO MEAN 1
 DEFINE TRAINING TO MEAN 2
 DEFINE ROTATING TO MEAN 3
 DEFINE SEPARATING TO MEAN 4
 DEFINE OJT.UPGRADE TO MEAN 6
 DEFINE ASSIGN TO MEAN 7
 DEFINE RETRAINING TO MEAN 8

DEFINE STATIC.INITIALIZATION,READ.MISSIONS,READ.BASE.OUTPUTS,
     MANPOWER.REQUIREMENTS,MISSION.SKILLS,MISSION.SUPPORT.SKILLS,
     BASE.SUPPORT.STAFF,TECH.TRAINING,BASIC.TRAINING,AIRCREW.TRAINING,
     TRAINING.SCHEDULER,YEAR.PLAN,TRAIL.AUTHORIZATION,PREDICT.SEPS,
     SKILL.ADJUSTMENT,RETENTION.ADJUSTMENT,UPGRADE.PLANNING,
     INCR.YEARGROUPS,SPREAD,AGGREGATE.PLANNING,DYNAMIC.INITIALIZATION,
     INIT.PERIODS,INIT.PERIOD0,CONFIGURE,
     DUPLICATE AS RELEASABLE ROUTINES
END
```

```
MAIN

CREATE AIRFORCE

CALL STATIC.INITIALIZATION

  PRINT 1 LINE THUS
+++++++END OF STATIC INITIALIZATION+++++++

  RELEASE STATIC.INITIALIZATION,READ.MISSIONS,READ.BASE.OUTPUTS
CALL MANPOWER.REQUIREMENTS

  PRINT 1 LINE THUS
########END OF MANPOWER REQUIREMENTS########

  RELEASE MANPOWER.REQUIREMENTS,MISSION.SKILLS,MISSION.SUPPORT.SKILLS,
     BASE.SUPPORT.STAFF,TECH.TRAINING,BASIC.TRAINING,AIRCREW.TRAINING
CALL INIT.YEAR0

  PRINT 1 LINE THUS
&&&&&&&&END OF INIT YEAR0&&&&&&&&

CALL AGGREGATE.PLANNING(YRS.TO.SIMULATE)

  PRINT 1 LINE THUS
@@@@@@@@END OF AGGREGATE PLANNING@@@@@@@@

  RELEASE AGGREGATE.PLANNING,TRAINING.SCHEDULER,YEAR.PLAN,
     TRIAL.AUTHORIZATION,PREDICT.SEPS,SKILL.ADJUSTMENT,
     RETENTION.ADJUSTMENT,UPGRADE.PLANNING,INCR.YEARGROUPS,
     SPREAD
CALL INSTALL.BASE.PERSONNEL
CALL DYNAMIC.INITIALIZATION(H.MONTHS)

  PRINT 1 LINE THUS
$$$HEREWEGO$$$$$

  RELEASE DYNAMIC.INITIALIZATION,INIT.PERIODS,INIT.PERIOD0,
     CONFIGURE,DUPLICATE
SCHEDULE AN INDUCTION NOW
SCHEDULE AN ASSIGNMENT NOW
SCHEDULE A SAVE.ASSIGN NOW
SCHEDULE A MOVEMENTS NOW
SCHEDULE A PERSONNEL.SAVING NOW
SCHEDULE A STORAGE.MANAGER AT 1

SCHEDULE A MASSIVE.RIF IN N.PERIOD UNITS
START SIMULATION
END
```

```
''
''STATIC INITTALIZATION MODULE
''
''    (01)-CONSTANTS FOR THE RUN (N.WHATEVER,ETC.)
''    (02)-"SKILLNAMES"              *
''    (03)-SKILL NAMES               (4 X,5 A 10)
''    (04)-"LEVELNAMES"              *
''    (05)-LEVEL NAMES               (4 X,A 10)
''    (06)-"BASEDATA"                *
''    (07)-NAME,STATE,MAJCOM         FREE FORM ALPHA
''    (08)-"TRAINTIMES"              *
''    (09)-TRAINING TIMES            FREE FORM
''    (10)-"SCHOOLDATA"              *
''    (11)-OTS.TIME, BMT.TIME        FREE FORM
'' (11.1)-"MISSIONS"
'' (11.2)-MISSION DATA               FREE FORM (1 CARD PER MISSION)
''                                              +SKILL.COEFF
'' (11.3)-"OUTPUTS"
'' (11.4)-OUTPUTS BY YEAR, BASE, (S,UTIL.FACTOR) FF
'' (11.5)-"YEARGRPFRAC"
'' (11.6)-OFF. & AIR.INITIAL.YG.PCT   FF
''    (12)-"YGSPLITS"                *
''    (13)-YEAR GRP SPLITS           FF
''    (14)-"RETENTION"               *
''    (15)-RETENTION RATES           FF
'' (15.1)-"EXITFRACT"
'' (15.2)-EXIT FRACTION   FF
''    (16)-"LSPLIT"                  *
''    (17)-DESIRED LSPLIT            FF
''    (18)-"TRAVELTIME"              *
''    (19)-TRAVEL TIMES              FF
''    (20)-"PIPECAPAC"               *
''    (21)-PIPELINE CAPACITY         FF
''    (22)-"MINMANFRAC"              *
''    (23)-MIN.MANPOWER.FRAC         FF
''    (26)-"OJTPCT"                  *
''    (27)-OJT %S                    FF
''    (28)-ROTATION DISTRIBUTION (BY ROTATION CYCLE)
''    (29)-IN.SHARE
''    (30)-IP.SHARE
''    (31)-AIR.OFF.EQUIV
''          * - HEADR FOR VERIFICATION
''          FF - FREE FORM INPUT
''
```

```
  ROUTINE FOR STATIC.INITIALIZATION
  DEFINE NS TO MEAN N.TECH.SCHOOLS
  DEFINE NP TO MEAN N.TRAVELTIME
  DEFINE ISEM.RUN.DATA TO MEAN 5 **INPUT UNIT FOR THIS DATA
  DEFINE B,S,L,MS,I,J,K,DAYS AS INTEGER VARIABLES
  DEFINE V,JUNK AS REAL VARIABLES
  DEFINE INTEGERDATA AS A 1-DIM INTEGER ARRAY
  DEFINE REALDATA AS A 1-DIM REAL ARRAY
  DEFINE ALPHADATA AS A 1-DIM ALPHA ARRAY
**
USE ISEM.RUN.DATA FOR INPUT
**CONSTANTS INPUT
  READ N.SKILL, N.AIRMEN.SKILLS,
     N.BASE, N.SCHOOL, YRS.TO.SIMULATE, MAX.LAG, H.MONTHS,
     AIR.AUTH.TRAJECTORY,OFF.AUTH.TRAJECTORY,AIR.INIT.AUTH.PCT,
     OFF.INIT.AUTH.PCT, MONTHLY.MAN.HRS, MANPOWER.PF
  LET OFFICER.SKILL1=N.AIRMEN.SKILLS+1

  LET N.LEVEL=5
  LET N.YEAR=YRS.TO.SIMULATE + PLAN.YEAR(MAX.LAG) + 1
  LET N.PERIOD=YRS.TO.SIMULATE*12+1
  LET YEAR0=N.YEAR
  LET PERIOD0=N.PERIOD
  PRINT 1 LINE WITH N.BASE,N.SKILL,N.LEVEL,OFF.INIT.AUTH.PCT AS FOLLOWS
N.BASE,SKILL,LEVEL = **** **** ****  OFF.INIT.AUTH.PCT = ****.****

**CREATE SKILLS, LEVELS, READ NAMES.
  LET N.LEN.TEXT=1
  CALL READCHECK("SKILLNAMES")
  RESERVE SKILL.NAME AS N.SKILL BY N.LEN.TEXT
  RESERVE ALPHADATA AS N.LEN.TEXT
  FOR EVERY SKILL, DO
     START NEW CARD
     READ ALPHADATA(1)
     LET SKILL.NAME(SKILL,1)=ALPHADATA(1)
     LOOP
  CALL READCHECK("LEVELNAMES")
  RESERVE OFFICER.NAME AND AIRMEN.NAME AS N.LEVEL
  FOR EACH LEVEL CALLED L,
  READ OFFICER.NAME(L), AIRMEN.NAME(L)
       AS S 4,A 10,S 1,A 10,/
**
```

```
**BASES
  CALL READCHECK("BASEDATA")
  READ APOE.EAST,APOE.WEST,BASICB,TTRB,UPTB
**RESERVE STORAGE TO HOLD  INVARIANT  BASE PROPERTIES:
  RESERVE NAME, CONT.LOCATION, SEP.POINT, OVERSEAS, BASE.TYPE,
          STATE.LOCATION, MAJCOM, P.BS.PS, S.BS.PS, M.BS.PS AND
          ROTATION.CYCLE AS N.BASE
  FOR EVERY BASE DO
   START NEW CARD
      SKIP 1 FIELD
      READ NAME(BASE)
      SKIP 2 FIELDS
      READ STATE.LOCATION(BASE),MAJCOM(BASE),BASE.TYPE(BASE),
           ROTATION.CYCLE(BASE)
      IF BASE.TYPE(BASE) NE 3 AND BASE.TYPE(BASE) NE 4
         FILE THIS BASE IN BASE.PRIORITY.SET(AIRFORCE)
         REGARDLESS
      IF MAJCOM(BASE) = "USAFE"
      LET CONT.LOCATION(BASE)=USAFE
      LET SEP.POINT(BASE)=APOE.EAST
      LET OVERSEAS(BASE)=1
 ELSE IF MAJCOM(BASE) = "PACAF"
      LET CONT.LOCATION(BASE)=PACAF
      LET SEP.POINT(BASE)=APOE.WEST
      LET OVERSEAS(BASE)=1
  ELSE
      LET CONT.LOCATION(BASE)=CONUS
      LET SEP.POINT(BASE)=BASE
      LET OVERSEAS(BASE)=0
  ALWAYS ALWAYS
      LOOP
```

```
**SCHOOLS. BUILD THEM FROM TRAINING TIME MATRIX.
**USING NON-ZERO ELEMENTS.
**RESERVE STORAGE FOR INVARIANT SCHOOL DATA
  RESERVE DURATION, SKILL.TAUGHT, LEVEL.TAUGHT, LEVEL.ENTER,
     BASE.LOC, AND SCHOOL.CAPACITY  AS N.SCHOOL
  CALL READCHECK("TRAINTIMES")
  RESERVE REALDATA AS 3
  RESERVE SCHOOL.CHART AS N.SKILL.TYPES BY N.LEVEL
  RESERVE TRAINING.TIME AS N.SKILL.TYPES BY N.LEVEL
  LET NS=0
  WRITE N.SCHOOL,N.SKILL.TYPES AS /," SCHOOLS, SKILLS ",2 I 5,/
  FOR S=1 TO N.SKILL.TYPES DO
**CREATING TECH SCHOOLS ONLY.
**WILL BE READING TT FOR LEVELS 2,3,4 (OUTCOME)
**1 AND 5 WILL BE ZERO.
     LET TRAINING.TIME(S,1)=0.
     LET TRAINING.TIME(S,5)=0.
     READ REALDATA
     FOR J=1 TO 3, DO
    IF REALDATA(J) NE 0.0 OR (((S >= PILOTF AND S <= PILOTL) OR
    (S >= NAVF AND S <=NAVL)) AND J EQ 1)
        LET NS=NS+1
        LET DURATION(NS)=REALDATA(J)
        LET SKILL.TAUGHT(NS)=S
        LET LEVEL.TAUGHT(NS)=J+1
        LET LEVEL.ENTER(NS)=J
     REGARDLESS
        LET TRAINING.TIME(S,J+1)=REALDATA(J)
        LET SCHOOL.CHART(S,J+1)=NS
     LOOP
     LOOP
**REST OF SCHOOL DATA BY SCHOOL, ALSO BMT, OTS.
  CALL READCHECK("SCHOOLDATA")
  READ OTS.TIME,BMT.TIME
  LET BMT=NS+1
  LET OTS=NS+2
  FOR S = 1 TO NS, DO
     LET I= SKILL.TAUGHT(S)
     IF I < N.ATRMEN.SKILLS OR I > NAVL
        LET BASE.LOC(S)=TTRB
     ELSE
        LET BASE.LOC(S)=UPTR
     ALWAYS
     LOOP
  LET BASE.LOC(BMT)=BASICB
  LET BASE.LOC(OTS)=BASICB
  LET DURATION(BMT)=BMT.TIME
  LET DURATION(OTS)=OTS.TIME
  LET LEVEL.TAUGHT(BMT)=1
  LET LEVEL.TAUGHT(OTS)=1
```

```
**TRAINING CONSTANTS--NO WASHOUT RATE FOR NOW
  RESERVE TRAINING.CONSTANT AS N.SKILL
  FOR S=1 TO N.AIRMEN.SKILLS
     LET TRAINING.CONSTANT(S)=BMT.TIME+TRAINING.TIME(S,LEVEL3)
  FOR S=OFFICER.SKILL1 TO N.SKILL,
     LET TRAINING.CONSTANT(S)=OTS.TIME+TRAINING.TIME(S,LEVEL3)



  CALL READCHECK("MISSIONS")
  CALL READ.MISSIONS


**INITIALIZE YEARS AND 2 SETS OF YEAR GROUPS
  CREATE EACH YEAR
  LET YG.ARRAY(*,*,*)=0
  RESERVE YG.ARRAY(*,*,*) AS N.SKILL BY N.LEVEL BY N.YEAR.GROUP+1
  LET YEAR.GROUPS(YEAR0)=YG.ARRAY(*,*,*)
  LET YG.ARRAY(*,*,*)=0
  RESERVE YG.ARRAY AS N.SKILL BY N.LEVEL BY N.YEAR.GROUP+1
  LET YEAR.GROUPS(1)=YG.ARRAY(*,*,*)


  CALL READCHECK("OUTPUTS")
  CALL READ.BASE.OUTPUTS


**YEAR GROUP INITIALIZATION FRACTIONS BY YG,L
  RESERVE OFF.INITIAL.YG.PCT(*,*),
          AIR.INITIAL.YG.PCT(*,*)  AS N.YEAR.GROUP BY N.LEVEL
  CALL READCHECK("YEARGRPERC")
  RELEASE REALDATA
  RESERVE REALDATA AS 4
  FOR YG=YG1 TO YG30, DO
     READ REALDATA
     FOR L=1 TO 4,
        LET OFF.INITIAL.YG.PCT(YG,L)=REALDATA(L)
     LOOP
  RELEASE REALDATA
  RESERVE REALDATA AS 5
  FOR YG=YG1 TO YG30, DO
     READ REALDATA
     FOR EACH LEVEL CALLED L,
        LET AIR.INITIAL.YG.PCT(YG,L)=REALDATA(L)
     LOOP
```

```
**YEAR GROUPS SPLIT FACTORS FOR UPGRADES.
  CALL READCHECK("YGSPLITS")
  RESERVE YG.SPLIT AS N.SKILL.TYPES BY *
**OFFICER YEAR GROUP SPLITS
  RESERVE YG.SPLIT(OFFICER.SKILL1,*,*) AS N.LEVEL BY
     N.YEAR.GROUP
  FOR S=OFFICER.SKILL1+1 TO N.SKILL,
     LET YG.SPLIT(S,*,*)=YG.SPLIT(OFFICER.SKILL1,*,*)
  FOR I=YG1 TO YG30, FOR L=LEVEL1 TO LEVEL7
     READ YG.SPLIT(OFFICER.SKILL1,L,I)
**AIRMEN YEAR GROUP SPLITS.
  RESERVE YG.SPLIT(1,*,*) AS N.LEVEL BY N.YEAR.GROUP
  FOR I=2 TO N.AIRMEN.SKILLS
     LET YG.SPLIT(I,*,*)=YG.SPLIT(1,*,*)
  FOR I=YG1 TO YG30, DO
     READ JUNK
     FOR L=LEVEL1 TO LEVEL7, DO
        READ YG.SPLIT(1,L,I)
        LOOP
     LOOP



**RETENTION RATES
  CALL READCHECK("RETENTION")
  RESERVE RETENTION.RATE AS N.SKILL.TYPES BY *
**AIRMEN RETENTION RATES..
  RESERVE RETENTION.RATE(1,*,*) AS N.LEVEL BY N.YEAR.GROUP
  FOR I=2 TO N.AIRMEN.SKILLS,
     LET RETENTION.RATE(I,*,*)=RETENTION.RATE(1,*,*)
  FOR I=YG1 TO YG30, FOR L=LEVEL1 TO LEVEL9
     READ RETENTION.RATE(1,L,I)
**OFFICER RETENTION RATES
  RESERVE RETENTION.RATE(OFFICER.SKILL1,*,*) AS N.LEVEL BY
     N.YEAR.GROUP
  FOR S=OFFICER.SKILL1+1 TO N.SKILL,
     LET RETENTION.RATE(S,*,*)=RETENTION.RATE(OFFICER.SKILL1,*,*)
  FOR I=YG1 TO YG30, FOR L=LEVEL1 TO LEVEL9
     READ RETENTION.RATE(OFFICER.SKILL1,L,I)

**RETENTION VARIANCES  ARE ALL 1.0
  RESERVE RETENTION.VARIANCE AS N.SKILL BY N.LEVEL
  FOR EVERY SKILL, FOR EVERY LEVEL, LET RETENTION.VARIANCE=1.0

**EXIT FRACTION
  CALL READCHECK("EXITFRACT")
  RESERVE EXIT.F AS  12
  READ EXIT.F
```

```
**DESIRED LSPLIT. READ FOR YEAR 1 AND PERPETUATE THRU YEARS.
  CALL READCHECK("LSPLIT")
  FOR I=2 TO N.YEAR, DO
     RELEASE DESIRED.LSPLIT(I,*,*)
     LET DESIRED.LSPLIT(I,*,*)=DESIRED.LSPLIT(1,*,*)
     LOOP
  FOR S=1 TO N.SKILL, FOR L=LEVEL3 TO LEVEL9
     READ  DESIRED.LSPLIT(1,S,L)   **N.B. LSPLIT(1,S,1)=0.0 ALWAYS



**PIPELINES. INITIALIZE FROM PIPE CHAART.
  RESERVE PIPE.CHART AS N.BASE BY N.BASE
  CALL READCHECK("TRAVELTIME")
  LET NP=0   **WILL BE # OF PIPES TO CREATE.
  FOR I=1 TO N.BASE, ALSO FOR J=1 TO N.BASE, DO
     READ DAYS
     IF DAYS NE 0
        LET NP=NP+1
     REGARDLESS
        LET PIPE.CHART(I,J)=DAYS
     LOOP
  CREATE EVERY TRAVELPIPE
  LET K=0
  FOR I=1 TO N.BASE, ALSO FOR J=1 TO N.BASE,DO
     IF PIPE.CHART(I,J) NE 0
        LET K=K+1
        LET MIN.TRAVEL.TIME(K)=PIPE.CHART(I,J)/30.
        LET PIPE.CHART(I,J)=K
        LET EXIT.BASE(K)=J
     REGARDLESS
     LOOP
  FOR EACH BASE CALLED I, WITH OVERSEAS(I)=0
     AND I NE APOE.EAST AND I NE APOE.WEST, DO
     FOR EACH BASE CALLED J, WITH OVERSEAS(J)=1, DO
        IF CONT.LOCATION(J)=USAFE
           LET PIPE.CHART(I,J)=PIPE.CHART(I,APOE.EAST)
           LET PIPE.CHART(J,I)=PIPE.CHART(J,APOE.EAST)
        ELSE
           LET PIPE.CHART(I,J)=PIPE.CHART(I,APOE.WEST)
           LET PIPE.CHART(J,I)=PIPE.CHART(J,APOE.WEST)
        ALWAYS
        LOOP
     LOOP
  CALL READCHECK("PIPECAPAC")
  WRITE NP AS /," N.TRAVEL.PIPES = ",I 5,/
     FOR EVERY TRAVELPIPE CALLED I DO
     READ CAPACITY(I)
     LET CAPACITY(I)=CAPACITY(I)*30.*MIN.TRAVEL.TIME(I)
     LOOP
```

```
''MIN.MANPOWER.FRAC
  RESERVE MIN.MANPOWER.FRAC AS N.SKILL BY N.LEVEL
  FOR EVERY SKILL, FOR EVERY LEVEL, LET MIN.MANPOWER.FRAC(SKILL,LEVEL)=
        0.67
  CALL READCHECK("MINMANFRAC")
''READ INDIVIDUAL CHANGES TO THE MATRIX
     READ S,L,V
     WHILE S GT 0, DO
        LET MIN.MANPOWER.FRAC(S,L)=V
        READ S,L,V
        LOOP



''OJT PERCENTAGES
  RESERVE OJT.PCT AS N.SKILL BY N.LEVEL
''INITIALIZE LEVEL1=0.0, LEVEL 2-5 = 1.0.
  FOR S=1 TO N.SKILL, DO
     LET OJT.PCT(S,1)=0.
     FOR L=2 TO N.LEVEL, DO
        LET OJT.PCT(S,L)=1.0
        LOOP
     LOOP
  CALL READCHECK("OJTPCT")
''READ CHANGES TO THE MATRIX.
  READ S,L,V
  WHILE S GT 0, DO
     LET OJT.PCT(S,L)=V
     READ S,L,V
     LOOP
```

```
**OJT DELAY,IN MONTHS, FOR AN INELIGIBLE TO BECOME ELIGIBLE
  RESERVE OJT.DELAY AS N.LEVEL
  FOR L=1 TO 5, LET OJT.DELAY(L)=1
  LET OJT.DELAY(LEVEL3)=8
  LET OJT.DELAY(LEVEL5)=12
**READ ROTATION.DISTRIBUTION
  RESERVE INITIAL.ROTATION.PCT AS 3 BY 24
  LET REALDATA(*)=0
  RESERVE REALDATA AS 3
  FOR I=1 TO 24,DO
     READ REALDATA
     FOR J=1 TO 3, DO
        LET INITIAL.ROTATION.PCT(J,I)=REALDATA(J)
        LOOP
     LOOP
**READ IP.SHARE, IN.SHARE
  RESERVE IN.SHARE AS 8
  RESERVE IP.SHARE AS 7
  RESERVE AIR.OFF.EQUIV AS 51
  READ IN.SHARE
  READ IP.SHARE
  READ AIR.OFF.EQUIV
**
  RELEASE REALDATA,INTEGERDATA,ALPHADATA


  RETURN
  END  **OF STATIC INITIALIZATION.........
```

```
ROUTINE TO READ.BASE.OUTPUTS
  DEFINE T AS AN INTEGER VARIABLE
  LET YEAR=0
  UNTIL YEAR=YEARO DO....
     START NEW CARD
     READ YEAR
     IF YEAR=0 LET YEAR=YEARO ALWAYS
     LET BASE=0
     UNTIL BASE=N.BASE DO
        START NEW CARD
        READ BASE,T
        WHILE T NE 0 DO..
           CREATE AN OUTPUT
           LET TYPE=T
           READ QUANTITY(OUTPUT), UTILIZATION(OUTPUT)
           FILE OUTPUT IN OUTPUTS(YEAR,BASE)
           READ T
           LOOP
        LOOP
     LOOP
END
```

```
ROUTINE TO READ.MISSIONS
  DEFINE SC AS A 1-DIM REAL ARRAY
  DEFINE S AS AN INTEGER VARIABLE
  START NEW CARD
  READ N.MISSION,UPT.CAP.FACTOR,UNT.CAP.FACTOR,TT.CAP.FACTOR,
       RMT.CAP.FACTOR,OTS.CAP.FACTOR
  CREATE EACH MISSION
  LET MISSION=0
  UNTIL MISSION=N.MISSION DO
     START NEW CARD
     READ MISSION
     READ STD.EQUIPMENT(MISSION),STD.FLYING.HRS(MISSION),
          MH.PER.FH(MISSION),MUNITIONS.LF(MISSION),
        FLT.PLAN.COEFF(MISSION), STD.MUNITIONS(MISSION)
     LET SC(*)=1
     RESERVE SC(*) AS N.SKILL.TYPES
     LET SKILL.COEFF(MISSION)=SC(*)
     START NEW CARD
     READ S
     WHILE S NE 0 DO
        READ SC(S)
        READ S
        LOOP
     LOOP
END
```

```
''
  ROUTINE FOR MANPOWER.REQUIREMENTS
''
  DEFINE TEMP,MP,MP.TOTAL AS 1-DIM INTEGER SAVED ARRAYS
  DEFINE Y,B,S,L,FLAG.9,FLAG.12,F,E,M,FLYING.TRAINEES,A.TECH,
         O.TECH,FP.TOTAL,EQUIP.TOTAL,MUNITIONS.TOTAL
         AS INTEGER VARIABLES
  DEFINE MISSION.POPULATION AS AN INTEGER VARIABLE
  DEFINE BASE.NAVS,BASE.PILOTS,TOTAL.NAVS,TOTAL.PILOTS
        AS INTEGER VARIABLES

  RESERVE TEMP AS N.SKILL.TYPES
  RESERVE MP AND MP.TOTAL AS N.SKILL
''
  FOR EACH YEAR CALLED Y, DO
     LET TOTAL.PILOTS=0
     LET TOTAL.NAVS=0
     LET MISSION.POPULATION=0
     LET FLYING.TRAINEES=0
     LET A.TECH=0
     LET O.TECH=0
     FOR EACH SKILL CALLED S DO
       LET TEMP(S)=0
       LET MP(S)=0
       LET MP.TOTAL(S)=0
     LOOP
 ''
''BASES RANKED IN SET...OPS BASES, THEN UPT BASE.
 ''
     FOR EACH B IN BASE.PRIORITY.SET, DO
        LET FLAG.9=0 ''FLAG IF OUTPUT TYPE 9 PRESENT
        LET FLAG.12=0 ''FLAG IF OUTPUT TYPE 12 PRESENT
        FOR EACH O IN OUTPUTS(Y,B), DO
           IF TYPE(O)=12 LET FLAG.12=0
           ELSE IF TYPE(O)=9 LET FLAG.9=0
           ELSE
              CALL MISSION.SKILLS GIVEN O,TEMP(*),TOTAL.PILOTS,
                                          TOTAL.NAVS
                                  YIELDING F,E,M,FLYING.TRAINEES
              ADD F TO FP.TOTAL
              ADD E TO EQUIP.TOTAL
              ADD M TO MUNITIONS.TOTAL
              FOR EACH SKILL CALLED S, DO
                 ADD TEMP(S) TO MP(S)
                 IF S>= PILOTF AND S <= PILOTL
                    ADD TEMP(S) TO BASE.PILOTS
                    ADD TEMP(S) TO TOTAL.PILOTS
                 ELSE IF S >= NAVF AND S <= NAVL
                    ADD TEMP(S) TO BASE.NAVS
                    ADD TEMP(S) TO TOTAL.NAVS
                 ALWAYS ALWAYS
                 LET TEMP(S)=0
                 LOOP
              ALWAYS ALWAYS
              LOOP
```

```
 ''
''IF MISSION SUPPORT SKILLS REQUESTED, CALCULATE THEM
 ''
            IF FLAG.9 NE 0
               CALL MISSION.SUPPORT.SKILLS(FLAG.9,TEMP(*),F,E,M,
                    BASE.PILOTS,BASE.NAVS,B)
               FOR EACH SKILL CALLED S, DO
                  ADD TEMP(S) TO MP(S)
                  LET TEMP(S)=0
                  LOOP
            REGARDLESS
 ''
''IF BASE SUPPORT STAFF REQUESTED, CALCULATE THEM
 ''
      FOR EACH SKILL CALLED S DO
         ADD MP(S) TO MISSION.POPULATION
      IF BASE.TYPE(B) = 2
         LET MISSION.POPULATION=MISSION.POPULATION+FLYING.TRAINEES
      REGARDLESS
            IF FLAG.12 NE 0
               CALL BASE.SUPPORT.STAFF(FLAG.12,TEMP(*),MISSION.POPULATION)
               FOR EACH SKILL CALLED S, DO
                  ADD TEMP(S) TO MP(S)
                  LET TEMP(S)=0
                  LOOP
            REGARDLESS
            CALL MP.SPLIT(MP(*),B,Y)
            FOR EACH SKILL CALLED S, DO
               ADD MP(S) TO MP.TOTAL(S)
               LET MP(S)=0
               LOOP
            LET MISSION.POPULATION=0
            LET BASE.PILOTS=0
            LET BASE.NAVS=0
            LOOP
         LOOP
 ''
''TECH TRAINING BASES
 ''
      FOR EACH BASE CALLED B, WITH BASE.TYPE(B)=3, DO
         CALL TECH.TRAINING GIVEN MP.TOTAL(*),TEMP(*),B,Y
                            YIELDING A.TECH,O.TECH
         CALL MP.SPLIT(TEMP(*),B,Y)
         FOR EACH SKILL CALLED S, DO
         ADD TEMP(S) TO MP.TOTAL(S)
         LET TEMP(S)=0
         LOOP
         LOOP
```

```
**BASIC TRAINING BASES
     FOR EACH BASE CALLED B WITH BASE.TYPE(B)=4, DO
        CALL BASIC.TRAINING GIVEN TEMP(*),FLYING.TRAINEES,
             O.TECH,A.TECH,B,Y
        CALL MP.SPLIT(TEMP(*),B,Y)
        FOR EACH SKILL CALLED S, DO
        ADD TEMP(S) TO MP.TOTAL(S)
        LET TEMP(S)=0
        LOOP
     RELEASE MP.TOTAL
     RESERVE MP.TOTAL AS N.SKILL
     LOOP
  LOOP
  RETURN
  END ** OF MANPOWER.REQUIREMENTS
```

```
''
  ROUTINE FOR MP.SPLIT GIVEN     MP, ''TOTAL POP FOR THIS BASE BY SK
                                 B,  ''BASE INDEX
                                 Y   ''YEAR INDEX
''
  DEFINE MP AS A 1-DIM INTEGER ARRAY
  DEFINE T,U AS 2-DIM INTEGER ARRAYS
  DEFINE I,B,Y,S,L AS INTEGER VARIABLES
''SPLIT BASES POP BY SKILL INTO YE.MANPOWER.DESIRED, YE.MIN MANPOWER
  RESERVE T AND U AS N.SKILL BY *
  LET YE.MANPOWER.DESIRED(Y,B)=T(*,*)
  LET YE.MIN.MANPOWER(Y,B)=U(*,*)
  FOR EACH SKILL CALLED S WITH MP(S) NE 0, DO
     RESERVE T(S,*) AS N.LEVEL
     RESERVE U(S,*) AS N.LEVEL
     LET I=MP(S)
     FOR EACH LEVEL CALLED L, DO
        LET T(S,L)=I*DESIRED.LSPLIT(Y,S,L)
        LET U(S,L)=T(S,L)*MIN.MANPOWER.FRAC(S,L)
        ADD T(S,L) TO USAF.MANPOWER.DESIRED(Y,S,L)
        ADD U(S,L) TO USAF.MIN.MANPOWER(Y,S,L)
        LOOP
     LOOP
  RETURN
  END ''OF MP.SPLIT
```

```
''
  ROUTINE FOR MISSION.SKILLS GIVEN     SQUADRON,REQ,M.PILOTS,M.NAVS
                                 YIELDING TOTAL.FP,TOTAL.EQUIP,MUNITIONS,
                                          TOTAL.FT
''
  DEFINE UTIL.FACTOR AS A REAL VARIABLE
  DEFINE REQ AS A 1-DIM INTEGER ARRAY
  DEFINE SC AS A 1-DIM REAL ARRAY
  DEFINE SQUADRON,TOTAL.EQUIP,TOTAL.FP,MUNITIONS,TOTAL.MAINT,
         AVIONICS.MAINT,FIELD.MAINT,ORG.MAINT,S,T,M.PILOTS,M.NAVS,
         FT,NT AS INTEGER VARIABLES
''
  LET T=TYPE(SQUADRON)
  IF T = 8
     CALL AIRCREW.TRAINING GIVEN M.PILOTS,M.NAVS,SQUADRON,T
                           YIELDING FT,NT
     LET TOTAL.FT=FT+NT
  REGARDLESS
  LET TOTAL.EQUIP=STD.EQUIPMENT(T)*QUANTITY(SQUADRON)
  LET TOTAL.FH=TOTAL.EQUIP*UTILIZATION(SQUADRON)  ''FLYING HOURS
  LET TOTAL.FP=FLT.PLAN.COEFF(T)*TOTAL.FH   ''FLIGHT PLANS
  LET TOTAL.MAINT=(TOTAL.FH*MH.PER.FH(T))/
                  (MONTHLY.MAN.HRS*MANPOWER.PE)
  LET MUNITIONS=TOTAL.EQUIP*MUNITIONS.LF(T)*STD.MUNITIONS(T)
  LET AVIONICS.MAINT=.232*TOTAL.MAINT
  LET FIELD.MAINT=.353*TOTAL.MAINT
  LET ORG.MAINT=.315*TOTAL.MAINT
  LET UTIL.FACTOR=TOTAL.FH/(STD.FLYING.HRS(T)*QUANTITY(SQUADRON))
  LET SC(*)=SKILL.CONSTANTS(T)
  FOR EACH SKILL CALLED S, WITH SC(S) NE 0, DO
     GO TO EQU(S)
''
''AIRCREW SKILLS...PILOTS, NAVIGATORS, AIRMEN, AIRCREW
'EQU(1)' 'EQU(2)' 'EQU(3)' 'EQU(51)' 'EQU(52)' 'EQU(53)' 'EQU(54)'
'EQU(55)' 'EQU(56)' 'EQU(57)' 'EQU(58)' 'EQU(59)' 'EQU(60)'
'EQU(61)' 'EQU(62)' 'EQU(63)' 'EQU(64)' 'EQU(65)' 'EQU(66)'
  LET REQ(S)=SC(S)*UTIL.FACTOR
     CYCLE
''AVIONICS MAINTENANCE
'EQU(17)' 'EQU(18)' 'EQU(19)' 'EQU(20)' 'EQU(21)' 'EQU(22)'
     LET REQ(S)=SC(S)*AVIONICS.MAINT
     CYCLE
''INSTRUMENT AND DEFENSIVE SYSTEMS TRAINERS
'EQU(23)' 'EQU(24)'
     LET REQ(S)=SC(S)*FT
     CYCLE
```

```
**BOMB/NAVIGATOR/TACTICS TRAINERS
'EQU(25)'
     LET REQ(S)=SC(S)*NT
     CYCLE
**FIELD MAINTENANCE
'EQU(27)' 'EQU(36)'
     LET REQ(S)=SC(S)*FIELD.MAINT
     CYCLE
**ORGANIZATIONAL MAINTENANCE
'EQU(28)' 'EQU(29)' 'EQU(30)' 'EQU(31)'
     LET REQ(S)=SC(S)*ORG.MAINT
     CYCLE
**WEAPONS MECHANIC
'EQU(33)'
     LET REQ(S)=SC(S)*TOTAL.EQUIP
     CYCLE
**MAINTENANCE OFFICER
'EQU(72)'
     LET REQ(S)=SC(S)*TOTAL.MAINT
     CYCLE
**MUNITIONS OFFICER
'EQU(73)'
   LET REQ(S)=SC(S)*SC(33)*TOTAL.EQUIP
     CYCLE
     LOOP
END
```

```
''
''
''
  ROUTINE FOR MISSION.SUPPORT.SKILLS GIVEN    SUPPORT,REQ,TOTAL.FP,
         TOTAL.EQUIP,MUNITIONS,BASE.PILOTS,BASE.NAVS,B
''
  DEFINE REQ AS A 1-DIM INTEGER ARRAY
  DEFINE SC AS A 1-DIM REAL ARRAY
  DEFINE SUPPORT,TOTAL.FP,TOTAL.EQUIP,MUNITIONS,BASE.PILOTS,BASE.NAVS,
         B,WLF,PILOT.EQUIV AS INTEGER VARIABLES
''
  LET WLF=(300+TOTAL.EQUIP)/5   ''WORK LOAD FACTOR
  LET PILOT.EQUIV=BASE.PILOTS+(BASE.NAVS/3)
  LET SC(*)=SKILL.CONSTANTS(TYPE(SUPPORT))
  FOR EACH SKILL CALLED S, WITH SC(S) NE 0, DO
  GO TO EQU(S)
''COMMUNICATIONS, ELECTRONICS AND RELATED FUNCTIONS
'EQU(4)' 'EQU(5)' 'EQU(10)' 'EQU(11)' 'EQU(14)' 'EQU(15)' 'EQU(26)'
     LET REQ(S)=SC(S)*WLF  CYCLE
''WEATHER EQUIPMENT REPAIR/OPERATION
'EQU(6)' 'EQU(12)'
     LET REQ(S)=SC(S)  CYCLE
''AIR OPERATIONS
'EQU(7)'
     IF TOTAL.FP <= 900
        LET REQ(S)=SC(S)
     ELSE
        LET REQ(S)=SC(S)+((TOTAL.FP-900)/200)
     ALWAYS
     IF PILOT.EQUIV <= 120
        LET REQ(S)=REQ(S)+1
     ELSE
        LET REQ(S)=REQ(S)+((PILOT.EQUIV+60)/180)
     ALWAYS CYCLE
''AIR TRAFFIC CONTROL
'EQU(8)'
     LET REQ(S)=SC(S)*REQ(7)  CYCLE
''DETECTION AND DEPLOYMENT
'EQU(9)'
     LET REQ(S)=SC(S)*REQ(3)  CYCLE
''RADAR EQUIPMENT REPAIR/MAINTENANCE
'EQU(13)'
     IF MAJCOM(3) = PACAF
        LET REQ(S)=SC(S)+4
     ELSE IF MAJCOM(8) = USAFE
        LET REQ(S)=SC(S)+10
     ELSE
        LET REQ(S)=SC(S)
     ALWAYS ALWAYS CYCLE
```

```
**MUNITIONS MAINTENANCE
'EQU(32)'
     IF MUNITIONS <= 1000
        LET REQ(S)=SC(S)
     ELSE
        LET REQ(S)=SC(S)+(((MUNITIONS-1000)/1000)*18)
     ALWAYS  CYCLE
**AIR TRAFFIC CONTROL OFFICER
'EQU(67)'
     LET REQ(S)=SC(S)*(REQ(7)+REQ(8))  CYCLE
**WEAPONS CONTROL OFFICER
'EQU(68)'
     LET REQ(S)=SC(S)*REQ(9)  CYCLE
**WEATHER OFFICER
'EQU(69)'
     LET REQ(S)=SC(S)*SC(5)  CYCLE
**COMMUNICATIONS/ ELECTRONICS OFFICER
'EQU(70)'
     LET REQ(S)=SC(S)*(REQ(10)+REQ(11)+REQ(12)+REQ(13)+REQ(14)+
                REQ(16)+REQ(26))
     CYCLE
**MUNITIONS OFFICER
'EQU(73)'
     LET REQ(S)=SC(S)*REQ(32)  CYCLE
**CARTOGRAPHIC OFFICER
'EQU(76)'
     LET REQ(S)=SC(S)*(SC(5)*WLF)  CYCLE
**INTELLIGENCE OFFICER
'EQU(85)'
     LET REQ(S)=SC(S)*(SC(4)*WLF)  CYCLE
     LOOP
END
```

```
''
  ROUTINE FOR AIRCREW.TRAINING GIVEN M.PILOTS,M.NAVS,TSQUAD,T
      YIELDING FLIGHT.TRAINEES,NAV.TRAINEES
''ESTIMATES FLIGHT AND NAVIGATOR TRAINEES AND REQUIRED INSTRUCTORS
  DEFINE M.PILOTS,M.NAVS,T,FLIGHT.TRAINEES,NAV.TRAINEES,INST.PILOTS,
         INST.NAVS,TSQUAD AS INTEGER VARIABLES
  DEFINE SC AS A 1-DIM REAL ARRAY
''
  LET INST.PILOTS=.124*M.PILOTS
  LET QUANTITY(TSQUAD)=.002*M.PILOTS
  LET FLIGHT.TRAINEES=.182*M.PILOTS
  LET INST.NAVS=.028*M.NAVS
  LET NAV.TRAINEES=.112*M.NAVS
''ESTABLISH SCHOOL CAPACITY FOR UNDERGRADUATE PILOT
''AND NAVIGATOR TRAINING
  LET SCHOOL.CAPACITY(SCHOOL.CHART(PILOTF,2))=FLIGHT.TRAINEES*
                UPT.CAP.FACTOR
  LET SCHOOL.CAPACITY(SCHOOL.CHART(NAVF,2))=NAV.TRAINEES*UNT.CAP.FACTOR
''CALCULATE STANDARD MANNING FOR INST.PILOTS
  LET SC(*)=SKILL.CONSTANTS(T)
  FOR PS=PILOTF TO PILOTL
     LET SC              (PS)=INST.PILOTS*IP.SHARE(PS-PILOTF+1)
''CALCULATE STANDARD MANNING FOR INST.NAVS
  FOR NS=NAVF TO NAVL
     LET SC              (NS)=INST.NAVS*IN.SHARE(NS-NAVF+1)
END
''
  ROUTINE FOR TECH.TRAINING GIVEN MISS.REQ,TECH.INST,B,Y
                    YIELDING AIR.TECH.TRAINEES,OFF.TECH.TRAINEES
''
  DEFINE MISS.REQ,TECH.INST AS 1-DIM INTEGER ARRAYS
  DEFINE TEMP AS A 1-DIM INTEGER SAVED ARRAY
  DEFINE SC AS A 1-DIM REAL ARRAY
  DEFINE B,Y,TECH.TRAINEES,TTB.POP,AIR.TECH.TRAINEES,
      OFF.TECH.TRAINEES AS INTEGER VARIABLES

  RESERVE TEMP AS N.SKILL.TYPES
```

```
..
  FOR EVERY OUTPUT IN OUTPUTS(Y,9), DO
     IF TYPE(OUTPUT)=10
        LET TTB.POP=0
        LET SC(*)=SKILL.CONSTANTS(TYPE(OUTPUT))
        FOR EACH SKILL CALLED S, WITH SC(S) NE 0 DO
           LET TECH.TRAINEES=(MISS.REQ(S)*SC(S))/
                 (1+((TT.CAP.FACTOR*SC(S)*TRAINING.TIME(S,2))/20))
           LET TECH.INST(S)=(TECH.TRAINEES*TT.CAP.FACTOR*
                 TRAINING.TIME(S,2))/20
           LET SCHOOL.CAPACITY(SCHOOL.CHART(S,2))=TECH.TRAINEES*
                 TT.CAP.FACTOR
             IF S<= N.AIRMEN.SKILLS
                ADD TECH.TRAINEES TO AIR.TECH.TRAINEES
             ELSE
                ADD TECH.TRAINEES TO OFF.TECH.TRAINEES
             ALWAYS
                LET TTB.POP=TTB.POP+TECH.INST(S)+TECH.TRAINEES
           LOOP
     ALWAYS
     IF TYPE(OUTPUT) = 12
        CALL BASE.SUPPORT.STAFF GIVEN OUTPUT,TEMP(*),TTB.POP
        FOR EACH SKILL CALLED S DO
          ADD TEMP(S) TO TECH.INST(S)
          LET TEMP(S)=0
          LOOP
        ALWAYS
     LOOP
  END
```

```
''
 ROUTINE FOR BASIC.TRAINING GIVEN    MT.INST,TOT.AIRCREW.TRAINEES,
        OFF.TECH.TRAINEES,AIR.TECH.TRAINEES,B,Y
''
  DEFINE MT.INST AS 1-DIM INTEGER ARRAYS
  DEFINE TEMP AS A 1-DIM INTEGER SAVED ARRAY
  DEFINE SC AS A 1-DIM REAL ARRAY
  DEFINE TOT.AIRCREW.TRAINEES,OFF.TECH.TRAINEES,AIR.TECH.TRAINEES,
        B,Y,BTB.POP,BM.TRAINEES,OTS.TRAINEES,OT.INST,BMT.INST
        AS INTEGER VARIABLES

  RESERVE TEMP AS N.SKILL.TYPES

  FOR EVERY OUTPUT IN OUTPUTS(Y,B), DO
     IF TYPE(OUTPUT)=11
        LET BTB.POP=0
        LET BM.TRAINEES=AIR.TECH.TRAINEES*1.06
        LET OTS.TRAINEES=TOT.AIRCREW.TRAINEES+OFF.TECH.TRAINEES
        LET OT.INST=(OTS.TRAINEES*OTS.CAP.FACTOR*OTS.TIME)/30
        LET BMT.INST=(BM.TRAINEES*BMT.CAP.FACTOR*BMT.TIME)/40
        LET SC(*)=SKILL.CONSTANTS(TYPE(OUTPUT))
        FOR S=1 TO N.AIRMEN.SKILLS, WITH SC(S) NE 0, DO
           LET MT.INST(S)=SC(S)*BMT.INST
           ADD MT.INST(S) TO BTB.POP
           LOOP
        FOR S=OFFICER.SKILL1 TO N.SKILL, WITH SC(S) NE 0, DO
           LET MT.INST(S)=SC(S)*OT.INST
           ADD MT.INST(S) TO BTB.POP
           LOOP
        LET BTB.POP=BTB.POP+BM.TRAINEES+OTS.TRAINEES
        LET SCHOOL.CAPACITY(OTS)=OTS.TRAINEES*OTS.CAP.FACTOR
        LET SCHOOL.CAPACITY(BMT)=BM.TRAINEES*BMT.CAP.FACTOR
     ALWAYS
  IF TYPE(OUTPUT)=12
     CALL BASE.SUPPORT.STAFF GIVEN OUTPUT,TEMP(*),BTB.POP
     FOR EACH SKILL CALLED S DO
        ADD TEMP(S) TO MT.INST(S)
        LET TEMP(S)=0
     LOOP
   ALWAYS
  LOOP
END
```

```
ROUTINE FOR BASE.SUPPORT.STAFF GIVEN SUPPORT,REQ,MISSION.POPULATION

  DEFINE REQ AS A 1-DIM INTEGER ARRAY
  DEFINE SC AS A 1-DIM REAL ARRAY
  DEFINE SUPPORT,MISSION.POPULATION,BASE.POP AS INTEGER VARIABLES
  DEFINE TOT.BASE.COEFF AS A REAL VARIABLE
  DEFINE AO TO MEAN AIR.OFF.EQUIV

  LET TOT.BASE.COEFF=0
  LET SC(*)=SKILL.CONSTANTS(TYPE(SUPPORT))

  FOR S=1 TO N.AIRMEN.SKILLS, WITH SC(S) NE 0 AND AO(S) NE 0,
     ADD SC(S)+(SC(S)*SC(AO(S))) TO TOT.BASE.COEFF
  ADD SC(89) TO TOT.BASE.COEFF

**SOLVE FOR BASE POPULATION
  LET BASE.POP=MISSION.POPULATION/(1.-TOT.BASE.COEFF)

**BASE SUPPORT REQUIREMENTS
  FOR S=1 TO N.AIRMEN.SKILLS-2, WITH SC(S) NE 0, DO
     LET REQ(S)=SC(S)*BASE.POP
     IF SC(AO(S)) NE 0
        LET REQ(AO(S))=REQ(AO(S))+SC(AO(S))*(SC(S)*BASE.POP)
     ALWAYS
     LOOP

**MEDICAL/DENTAL SUPPORT REQUIREMENTS
  LET REQ(50)=SC(50)*BASE.POP
  LET REQ(87)=.190*SC(87)*BASE.POP
  LET REQ(88)=.946*SC(88)*BASE.POP
  LET REQ(89)=SC(89)*BASE.POP
  LET REQ(90)=.219*SC(90)*SC(50)*BASE.POP
  LET REQ(91)=.145*SC(91)*SC(50)*BASE.POP

END
```

```
ROUTINE TO INIT.YEAR0
  DEFINE S,L,YG,R,C,LSIZE AS INTEGER VARIABLES

  LET YG.ARRAY(*,*,*)=YEAR.GROUPS(YEAR0)
  FOR EACH SKILL CALLED S, FOR EACH LEVEL CALLED L DO ...
     LET USAF.MISSION.AUTH(YEAR0,S,L)=USAF.MANPOWER.DESIRED(YEAR0,S,L)
     LET USAF.PROJECTION(YEAR0,S,L)=USAF.MANPOWER.DESIRED(YEAR0,S,L)
     LOOP
  FOR S=1 TO N.AIRMEN.SKILLS,ALSO FOR EACH LEVEL CALLED L, DO
     ADD USAF.PROJECTION(YEAR0,S,L) TO YES.AIRMEN(YEAR0)
     FOR YG=YG1 TO YG30
     LET YG.ARRAY(S,L,YG)=USAF.PROJECTION(YEAR0,S,L)*
                                  AIR.INITIAL.YG.PCT(YG,L)
     LOOP
  FOR S=OFFICER.SKILL1 TO N.SKILL,ALSO FOR EACH LEVEL CALLED L, DO
     ADD USAF.PROJECTION(YEAR0,S,L) TO YES.OFFICERS(YEAR0)
     FOR YG=YG1 TO YG30
     LET YG.ARRAY(S,L,YG)=USAF.PROJECTION(YEAR0,S,L)*
                                  OFF.INITIAL.YG.PCT(YG,L)
     LOOP
  RELEASE AIR.INITIAL.YG.PCT AND OFF.INITIAL.YG.PCT

  CALL YES.TRAJECTORY

  FOR EACH BASE DO...

     CALL DUPLICATE GIVEN YE.MANPOWER.DESIRED(YEAR0,BASE)
                    YIELDING YE.AUTHORIZATION(YEAR0,BASE)
     LOOP
END
```

```
ROUTINE FOR YES.TRAJECTORY
  DEFINE Y AS AN INTEGER VARIABLE
  DEFINE X AS A REAL VARIABLE

  FOR Y=1 TO N.YEAR, DO
     LET X=Y
     LET YES.AIRMEN(Y)=YES.AIRMEN(YEAR0)*(AIR.INIT.AUTH.PCT)*
   ((1.0+AIR.AUTH.TRAJECTORY)**(Y-1))
     LET YES.OFFICERS(Y)=YES.OFFICERS(YEAR0)*(OFF.INIT.AUTH.PCT)*
   ((1.0+OFF.AUTH.TRAJECTORY)**(Y-1))
     LOOP

END
```

```
ROUTINE FOR AGGREGATE.PLANNING GIVEN N.YEARS
DEFINE N.YEARS,MAX.YR,PN,Y AS INTEGER VARIABLES
LET PN=N.YEARS*12
LET MAX.YR=PLAN.YEAR(CEIL.F(REAL.F(PN+MAX.LAG)))

CALL YEAR.PLAN (YEAR0,1)  **YEAR0 IS SPECIAL
CALL PLAN.SAVING(1)

LET YEAR.GROUPS(2)=YEAR.GROUPS(YEAR0)
LET YEAR.GROUPS(YEAR0)=0
FOR Y=1 TO MAX.YR-1 DO....
  CALL YEAR.PLAN(Y,Y+1)
  CALL PLAN.SAVING(Y+1)
  LET YEAR.GROUPS(Y+2)=YEAR.GROUPS(Y)
  LET YEAR.GROUPS(Y)=0
  LOOP

FOR Y=1 TO MAX.YR, ALSO
   FOR B=1 TO N.BASE, WITH OVERSEAS(B)=0, DO
      LET X(*,*)=YE.AUTHORIZATION(Y,B)
     LET ISIZE=DIM.F(X(*,*))
      FOR S=1 TO ISIZE, DO
        LET JSIZE=LEN.F(X(S,*))
         FOR L=1 TO JSIZE, DO
            ADD X(S,L) TO CONUS.USAF.M.AUTH(Y,S,L)
            LOOP
        LOOP
      LOOP

  CALL RLSE.YEAR.PLAN.DATA

  CALL TRAINING.SCHEDULER (1,PN,1,MAX.YR) YIELDING
     USAF.HOLES(*,*,*),
     AIRMEN.RECRUITS(*),
     OFFICER.RECRUITS(*)

  RELEASE USAF.PCHANGE
END
```

```
ROUTINE FOR YEAR.PLAN GIVEN BASE.YR,NEXT.YR
  DEFINE BASE.YR,NEXT.YR,S,L AS INTEGER VARIABLES
  DEFINE AUTH0,AUTH1 AS 2-DIM REAL SAVED ARRAYS
  DEFINE NEW.RETEN AS A 3-DIM REAL SAVED ARRAY
RESERVE AUTH0,AUTH1 AS N.SKILL BY N.LEVEL
RESERVE NEW.RETEN AS  N.SKILL BY N.LEVEL BY N.YEAR.GROUP
CALL TRIAL.AUTHORIZATION GIVEN YES.OFFICERS(NEXT.YR),
                               YES.AIRMEN(NEXT.YR),
                               USAF.MANPOWER.DESIRED(NEXT.YR,*,*),
                  ''YIELDING'' AUTH0(*,*)
  CALL PR.YG.ARRAY GIVEN NEXT.YR
''PROJECT SEPARATIONS BASED ON STD RETENTION RATES.
    CALL PREDICT.SEPS GIVEN YEAR.GROUPS(BASE.YR),
                            RETENTION.RATE(*,*,*),
                            RETENTION.VARIANCE(*,*),
                ''YIELDING'' YEAR.GROUPS(NEXT.YR),
                            EXPECTED.SEPARATIONS(NEXT.YR,*,*),
                            ACTUAL.SEPARATIONS(NEXT.YR,*,*),
                            USAF.PROJECTION(NEXT.YR,*,*)
''
''ADJUST AUTH TO COVER MINIMUM SKILL REQUIREMENTS
    CALL SKILL.ADJUSTMENT GIVEN AUTH0(*,*),AUTH1(*,*),
                                USAF.MIN.MANPOWER(NEXT.YR,*,*),
                      ''YIELDING'' AUTH1(*,*)
  CALL PR.USAF.PROJ GIVEN NEXT.YR AND USAF.PROJECTION(NEXT.YR,*,*)
    CALL RETENTION.ADJUSTMENT GIVEN AUTH1(*,*),
                                  USAF.PROJECTION(NEXT.YR,*,*),
                                USAF.PROJECTION(BASE.YR,*,*),
                                  RETENTION.RATE(*,*,*),
                                  NEW.RETEN(*,*,*)
  CALL PR.USAF.PROJ GIVEN NEXT.YR AND USAF.PROJECTION(NEXT.YR,*,*)
''
''PROJECT SEPARATIONS BASED ON ADJUSTED RATES
    CALL PREDICT.SEPS GIVEN YEAR.GROUPS(BASE.YR),
                                NEW.RETEN(*,*,*),
                                RETENTION.VARIANCE(*,*),
                      ''YIELDING'' YEAR.GROUPS(NEXT.YR),
                                EXPECTED.SEPARATIONS(NEXT.YR,*,*),
                                ACTUAL.SEPARATIONS(NEXT.YR,*,*),
                                USAF.PROJECTION(NEXT.YR,*,*)
  CALL PR.USAF.PROJ GIVEN NEXT.YR AND USAF.PROJECTION(NEXT.YR,*,*)
''
''CALCULATE UPGRADE DEMAND
    CALL UPGRADE.PLANNING GIVEN AUTH1(*,*),NEXT.YR,
                                USAF.PROJECTION(NEXT.YR,*,*),
                     ''YIELDING'' USAF.SCHOOL.AUTH(NEXT.YR,*,*),
                                USAF.UPGRADES(NEXT.YR,*,*),
                                USAF.PCHANGE(NEXT.YR,*,*)
                        YIELDING AUTH1(*,*) ''ADJUSTED FOR TRAINING
''
```

```
**CALCULATE FINAL MISSION AUTHORIZATION, PROJECTION
FOR EACH SKILL CALLED S FOR L = 2 TO N.LEVEL DO
   LET USAF.MISSION.AUTH(NEXT.YR,S,L)=AUTH1(S,L)-
   USAF.SCHOOL.AUTH(NEXT.YR,S,L)
   ADD USAF.PCHANGE(NEXT.YR,S,L) TO USAF.PROJECTION(NEXT.YR,S,L)
   LOOP
**ADD EFFECTS OF UPGRADES AND RECRUITS TO YEAR GROUP MEMORY
CALL INCR.YEARGROUPS GIVEN USAF.UPGRADES(NEXT.YR,*,*),
              **YIELDING** YEAR.GROUPS(NEXT.YR)
 CALL PR.YG.ARRAY GIVEN NEXT.YR
**
**NOW USAF.MISSION.AUTH + USAF.SCHOOL.AUTH = USAF.PROJECTION = AUTH1
**             (L=LEVEL1 TO LEVEL9)
**DISTRIBUTE MISSION AUTHORIZATION TO BASES
FOR EACH BASE, DO
   CALL CONFIGURE GIVEN YE.MANPOWER.DESIRED(NEXT.YR,BASE)
                  YIELDING YE.AUTHORIZATION(NEXT.YR,BASE)
   CALL SPREAD GIVEN USAF.MISSION.AUTH(NEXT.YR,*,*),
                     YE.MANPOWER.DESIRED(NEXT.YR,BASE),
                     USAF.MANPOWER.DESIRED(NEXT.YR,*,*),
        **YIELDING** YE.AUTHORIZATION(NEXT.YR,BASE)
   LOOP
END
```

```
ROUTINE TO INCR.YEARGROUPS GIVEN UPGRADES,GROUP
  DEFINE MEN,EXTRA AS REAL VARIABLES
  DEFINE UPGRADES AS A 2-DIM REAL ARRAY
  DEFINE GROUP AS A 3-DIM INTEGER ARRAY
  DEFINE S,L,YG,CHANGE AS INTEGER VARIABLES
LET NEXTYR(*,*,*)=GROUP(*,*,*)
LET EXTRA=0
FOR EACH SKILL CALLED S DO
   WRITE S AS /," SKILL ",I 3
   FOR L BACK FROM LEVEL9 TO LEVEL3, FOR YG=YG1 TO YG30 DO
      LET MEN=UPGRADES(S,L)*YG.SPLIT(S,L-1,YG)+EXTRA
      LET EXTRA=FRAC.F(MEN)
      LET CHANGE=TRUNC.F(MEN)
      SUBTRACT CHANGE FROM NEXTYR(S,L-1,YG)
      ADD CHANGE TO NEXTYR(S,L,YG)
      WRITE L,YG,UPGRADES(S,L),YG.SPLIT(S,L-1,YG),MEN,EXTRA,CHANGE,
      NEXTYR(S,L-1,YG),NEXTYR(S,L,YG) AS /,2 I 3,D(5,1),D(6,3),2 D(6,2),
      3 I 5
      LOOP
   ADD TRUNC.F(UPGRADES(S,LEVEL1)) TO NEXTYR(S,LEVEL1,YG1)
   LOOP
END
```

```
ROUTINE TO PREDICT.SEPS GIVEN   THISYG,   ''CURRENT YEAR GROUP MEMORY
                                RATE,     ''RETENTION RATE
                                VARIANCE, ''RETENTION VARIANCE
                                NEXTYG,   ''NEXT YEAR GROUP
                                SEPS,     ''EXPECTED SEPARATIONS
                                TRUE.SEPS, ''ACTUAL SEPARATIONS
                                PROJECTED  ''EXPECTED POPULATION
''
  DEFINE VARIANCE AS A 2-DIM REAL ARRAY
  DEFINE THISYG,NEXTYG AS 3-DIM INTEGER ARRAYS
  DEFINE RATE AS A 3-DIM REAL ARRAY
  DEFINE SEPS,TRUE.SEPS,PROJECTED AS 2-DIM REAL ARRAYS
  DEFINE YG,S,L AS INTEGER VARIABLES
  DEFINE WILL.STAY,TRUE.RATE,EXTRA AS REAL VARIABLES
''
LET THISYR(*,*,*)=THISYG(*,*,*)
LET NEXTYR(*,*,*)=NEXTYG(*,*,*)
LET EXTRA=0
FOR EACH SKILL, FOR EACH LEVEL, DO
   LET SEPS(SKILL,LEVEL)=0
   LET TRUE.SEPS(SKILL,LEVEL)=0
   LET PROJECTED(SKILL,LEVEL)=0
   LOOP
FOR EACH SKILL CALLED S FOR EACH LEVEL CALLED L,
FOR YG=YG1 TO YG30, DO
''TRUE SEPARATIONS
   LET TRUE.RATE=MIN.F(1,RATE(S,L,YG)*VARIANCE(S,L))
   LET WILL.STAY=THISYR(S,L,YG)*TRUE.RATE+EXTRA
   LET NEXTYR(S,L,YG+1)=TRUNC.F(WILL.STAY)
   ADD THISYR(S,L,YG)-WILL.STAY TO TRUE.SEPS(S,L)
   LET EXTRA=FRAC.F(WILL.STAY)
''EXPECTED SEPARATIONS
   LET WILL.STAY=THISYR(S,L,YG)*RATE(S,L,YG)
   ADD THISYR(S,L,YG)-WILL.STAY TO SEPS(S,L)
   ADD WILL.STAY TO PROJECTED(S,L)
   LOOP
END
```

```
ROUTINE FOR RETENTION.ADJUSTMENT GIVEN A1,PROJECTED,PRIOR.POPULATION,
                                        RETEN,NEW.RETEN
  DEFINE RETEN,NEW.RETEN AS 3-DIM REAL ARRAYS
  DEFINE A1,PROJECTED,PRIOR.POPULATION AS 2-DIM REAL ARRAYS
  DEFINE RSUM AS A 2-DIM REAL SAVED ARRAY
  DEFINE HIGHER.NEEDS, NEWR AS REAL VARIABLES
  DEFINE S,L,YG AS INTEGER VARIABLES

  RESERVE RSUM AS N.SKILL BY N.LEVEL

  FOR EACH SKILL CALLED S, FOR EACH LEVEL CALLED L, DO....
     LET RSUM(S,L)=0
     ALSO FOR YG=YG1 TO YG30, DO
        LET NEW.RETEN(S,L,YG)=RETEN(S,L,YG)
        ADD RETEN(S,L,YG) TO RSUM(S,L)
        LOOP

**ADJUST NEW.RETEN IF PROJECTION EXCEEDS AUTHORIZATION
  FOR EACH SKILL CALLED S, DO....
     LET HIGHER.NEEDS=0
     FOR L BACK FROM LEVEL9 TO LEVEL3, DO
        IF A1(S,L)+HIGHER.NEEDS < PROJECTED(S,L)
           LET PROJECTED(S,L)=A1(S,L) + HIGHER.NEEDS
           LET NEWR=PROJECTED(S,L)/PRIOR.POPULATION(S,L)
           FOR YG=YG1 TO YG30
              LET NEW.RETEN(S,L,YG)=NEWR*RETEN(S,L,YG)/RSUM(S,L)
        REGARDLESS
        ADD A1(S,L)-PROJECTED(S,L) TO HIGHER.NEEDS
        LOOP
     LOOP
  RETURN
END
```

```
ROUTINE FOR SKILL.ADJUSTMENT  GIVEN  A0,OUTPUT.ARRAY,MIN.MANPOWER
                              YIELDING A1
  DEFINE A0,A1,OUTPUT.ARRAY AS 2-DIM REAL ARRAYS
  DEFINE MIN.MANPOWER AS A 2-DIM REAL ARRAY
  DEFINE S,L,SUM.MIN,TOTAL.MIN AS RECURSIVE INTEGER VARIABLES
  DEFINE SUM.L,TOTAL AS RECURSIVE REAL VARIABLES
LET A1(*,*)=OUTPUT.ARRAY(*,*)
FOR EACH SKILL CALLED S DO
   FOR EACH LEVEL CALLED L DO
      LET A1(S,L)=A0(S,L)
      ADD A0(S,L) TO SUM.L
      ADD MIN.MANPOWER(S,L) TO SUM.MIN
      LOOP
   IF SUM.L < SUM.MIN
      WRITE S,SUM.L,SUM.MIN AS /,"SKILL ",I 3," AUTHORIZATION IS"
            ,D(6,1)," --BELOW MIN OF",I 6
      REGARDLESS
   ADD SUM.L TO TOTAL
   ADD SUM.MIN TO TOTAL.MIN
   LET SUM.L=0
   LET SUM.MIN=0
   LOOP
IF TOTAL < TOTAL.MIN
   WRITE TOTAL,TOTAL.MIN AS /,/,"**** TOTAL AUTHORIZATION OF",
         D(8,1)," IS BELOW USAF MIN MANPOWER RQMT",D(8,1),
         "--ABORT",/
   ALWAYS
END
```

```
ROUTINE SPREAD GIVEN INPUT,NUMERATOR,DENOMINATOR,OUT
  DEFINE INPUT,DENOMINATOR AS 2-DIM REAL ARRAYS
  DEFINE OUT,NUMERATOR AS A 2-DIM INTEGER ARRAY
  DEFINE I,J,ISIZE,JSIZE AS INTEGER VARIABLES
  DEFINE EXTRA,MEN AS REAL VARIABLES
..
LET EXTRA=0
LET ISIZE=DIM.F(INPUT(*,*))
FOR I=1 TO ISIZE DO
   LET JSIZE=LEN.F(OUT(I,*))
   ALSO FOR J=2 TO JSIZE DO
   IF DENOMINATOR(I,J) NE 0.
      LET MEN=INPUT(I,J)*(NUMERATOR(I,J)/DENOMINATOR(I,J))
   ELSE
      LET MEN = 0
   ALWAYS
      ADD EXTRA TO MEN
      LET OUT(I,J)=TRUNC.F(MEN)
      LET EXTRA=FRAC.F(MEN)
      LOOP
END
```

```
ROUTINE TRAINING.SCHEDULER GIVEN P1,PN,  **FIRST & LAST DESIRED PERIOD
                               FIRST.YR.PLAN,
                               N.YRS.PLANNED
                          YIELDING HOLES,A.RECRUITS,O.RECRUITS
  DEFINE P1,PN,FIRST.YR.PLAN,N.YRS.PLANNED AS INTEGER VARIABLES
  DEFINE A.RECRUITS,O.RECRUITS AS 1-DIM INTEGER ARRAYS
  DEFINE HOLES AS A 3-DIM REAL ARRAY
  DEFINE S,L,Y,P,M,P2,LAST.YR.PLAN,MAX.P,YR.PLAN.P1,
  YR.PLAN.P12 AS INTEGER VARIABLES
  DEFINE BACKUP,RECRUITS,EXTRA,SEPS AS REAL VARIABLES
LET LAST.YR.PLAN=FIRST.YR.PLAN+N.YRS.PLANNED-1
LET MAX.P=12*LAST.YR.PLAN
RESERVE HOLES AS MAX.P BY N.SKILL BY N.LEVEL
FOR EACH SKILL CALLED S DO
**NO TRAINING OUT OF LEVEL9: ALL OJT OUT OF LEVEL7
   LET L=LEVEL9
   FOR Y BACK FROM LAST.YR.PLAN TO FIRST.YR.PLAN DO
      LET YR.PLAN.P12=12*Y
      LET YR.PLAN.P1=YR.PLAN.P12-11
      FOR P BACK FROM YR.PLAN.P12 TO YR.PLAN.P1 DO
         LET M=CALENDAR.MONTH(P)
         LET HOLES(P,S,L)=USAF.PCHANGE(Y,S,L)*EXIT.F(M)
         LET HOLES(P,S,L-1)=USAF.PCHANGE(Y,S,L-1)*EXIT.F(M)
         +HOLES(P,S,L)
         LOOP
      LOOP
   LET BACKUP=0
**HOLES IN LEVEL 5, 3, 1
   FOR L BACK FROM LEVEL5 TO LEVEL1 DO
      ADD TRAINING.TIME(S,L+1) TO BACKUP
      FOR P BACK FROM TRUNC.F(MAX.P-BACKUP)  TO P1 DO
         LET Y=PLAN.YEAR(P)
         LET M=CALENDAR.MONTH(P)
         LET P2=CEIL.F(P+TRAINING.TIME(S,L+1))
         IF Y >= FIRST.YR.PLAN
            LET SEPS=USAF.PCHANGE(Y,S,L)*EXIT.F(M)
         ELSE
            LET SEPS=0
         ALWAYS
         LET HOLES(P,S,L)=SEPS+
                          OJT.PCT(S,L+1)*HOLES(P,S,L+1)+
                          (1-OJT.PCT(S,L+1))*HOLES(P2,S,L+1)+
                          HOLES(P,S,L)
         LOOP
      LOOP
   LOOP
```

```
**RECRUITS REQUIRED FOR THIS UPGRADE PLAN = SUM OF ALL LEVEL1 HOLES
RESERVE A.RECRUITS AS PN
RESERVE O.RECRUITS AS PN
LET EXTRA=0
FOR EACH SKILL CALLED S, FOR P=P1 TO PN DO
   IF S <= N.AIRMEN.SKILLS
      LET RECRUITS=HOLES(CEIL.F(P+BMT.TIME),S,1)+
                   EXTRA
      ADD TRUNC.F(RECRUITS) TO A.RECRUITS(P)
      LET EXTRA=FRAC.F(RECRUITS)
   ELSE
      LET RECRUITS=HOLES(CEIL.F(P+OTS.TIME),S,1)+
                   EXTRA
      ADD TRUNC.F(RECRUITS) TO O.RECRUITS(P)
      LET EXTRA=FRAC.F(RECRUITS)
   ALWAYS
   LOOP
END
```

```
ROUTINE FOR TRIAL.AUTHORIZATION GIVEN YES.O,YES.A,DESIRED,AUTHO
  DEFINE YES.O,YES.A,DESIRE.O,DESIRE.A,S,L AS INTEGER RECURSIVE
 VARIABLES
  DEFINE DESIRED AS A 2-DIM REAL ARRAY
  DEFINE AUTHO AS A 2-DIM REAL ARRAY
  DEFINE PCT.O,PCT.A AS REAL VARIABLES
''
''
FOR S=1 TO N.AIRMEN.SKILLS, FOR EACH LEVEL CALLED L
   ADD DESIRED(S,L) TO DESIRE.A
FOR S=OFFICER.SKILL1 TO N.SKILL, FOR EACH LEVEL CALLED L
   ADD DESIRED(S,L) TO DESIRE.O
LET PCT.A=YES.A/DESIRE.A
LET PCT.O=YES.O/DESIRE.O
FOR S=1 TO N.AIRMEN.SKILLS, FOR EACH LEVEL CALLED L
   LET AUTHO(S,L)=DESIRED(S,L)*PCT.A
FOR S=OFFICER.SKILL1 TO N.SKILL, FOR EACH LEVEL CALLED L
   LET AUTHO(S,L)=DESIRED(S,L)*PCT.O
END
```

```
ROUTINE FOR UPGRADE.PLANNING GIVEN A,YR,PROJECTED,
                                   TO.SCHOOL,UPGRADES,CHANGE
                               YIELDING NEW.AUTH
  DEFINE A,TO.SCHOOL,UPGRADES,CHANGE,NEW.AUTH,PROJECTED
  AS 2-DIM REAL ARRAYS
  DEFINE S,L,YR AS INTEGER VARIABLES
  DEFINE T,PSUM,ASUM,UPGRDSUM AS REAL VARIABLES
''
LET NEW.AUTH(*,*)=A(*,*)
FOR EACH SKILL CALLED S DO
   LET PSUM=0
   LET UPGRDSUM=0
   LET ASUM=0
   FOR  L=LEVEL3 TO LEVEL9 DO
      ADD A(S,L) TO ASUM
      ADD PROJECTED(S,L) TO PSUM
WRITE S,L,A(S,L),PROJECTED(S,L) AS /,2 I 4,2 D(8,2)
      LOOP
''
''AN AMOUNT T COMES OUT OF AUTHORIZATION FOR LEVEL0 & 1 TRAINING
   LET T=(TRAINING.CONSTANT(S)*(ASUM-PSUM))/(12+TRAINING.CONSTANT(S))
WRITE S,TRAINING.CONSTANT(S),T AS I 4,S 1,2 D(8,2),/
   FOR L BACK FROM LEVEL9 TO LEVEL3 DO
      LET NEW.AUTH(S,L)=A(S,L)-T*DESIRED.LSPLIT(YR,S,L)
      LET CHANGE(S,L)=NEW.AUTH(S,L)-PROJECTED(S,L)
      ADD CHANGE(S,L) TO UPGRDSUM
      LET UPGRADES(S,L)=UPGRDSUM
      LET TO.SCHOOL(S,L-1)=UPGRADES(S,L)*(1-OJT.PCT(S,L))*
                TRAINING.TIME(S,L)/12
WRITE S,L,NEW.AUTH(S,L),CHANGE(S,L),UPGRADES(S,L),TO.SCHOOL(S,L),
   OJT.PCT(S,L),TRAINING.TIME(S,L) AS /,2 I 4,6 D(8,2),/
      LOOP
   LET UPGRADES(S,LEVEL1)=UPGRDSUM  ''LEVEL1 OUTPUT
   LOOP
END
```

```
ROUTINE PLAN.SAVING GIVEN YR
DEFINE YR AS AN INTEGER VARIABLE
  DEFINE SUPP1,SUPP2 AS 2-DIM INTEGER ARRAYS
  DEFINE Z AS AN INTEGER VARIABLE
  USE UNIT 3 FOR OUTPUT
LET Z=0
LET YEAR=YR
  WRITE YEAR AS "YEAR  ",I 2
FOR EACH SKILL,
  FOR EACH LEVEL, DO
     WRITE USAF.MISSION.AUTH(YEAR,SKILL,LEVEL),
           USAF.MANPOWER.DESIRED(YEAR,SKILL,LEVEL),
           EXPECTED.SEPARATIONS(YEAR,SKILL,LEVEL),
           USAF.UPGRADES(YEAR,SKILL,LEVEL) AS 2 D(7,0),2 D(5,0)
  LOOP
WRITE AS "BASE  "
FOR EACH BASE, DO
  LET SUPP1(*,*)=YE.MANPOWER.DESIRED
  LET SUPP2(*,*)=YE.AUTHORIZATION
  FOR EACH SKILL, ALSO
     FOR EACH LEVEL, DO
     IF LEN.F(SUPP1(SKILL,*))<LEVEL
        WRITE Z,Z AS 2 I 4
     ELSE
        WRITE SUPP2(SKILL,LEVEL),SUPP1(SKILL,LEVEL) AS 2 I 4
  ALWAYS LOOP LOOP
RETURN
END
```

```
ROUTINE TO RLSE.YEAR.PLAN.DATA
FOR EACH YEAR DO
LET Y=YEAR
  IF YEAR.GROUPS(YEAR) NE 0, RELEASE YEAR.GROUPS(YEAR) ALWAYS
  RELEASE USAF.PROJECTION(YEAR,*,*)
  RELEASE USAF.UPGRADES(YEAR,*,*)
  RELEASE USAF.SCHOOL.AUTH(YEAR,*,*)
  RELEASE USAF.MANPOWER.DESIRED(YEAR,*,*)
  RELEASE  USAF.MIN.MANPOWER(YEAR,*,*)
  LOOP
RELEASE RETENTION.RATE(1,*,*)
RELEASE RETENTION.RATE(OFFICER.SKILL1,*,*)
RELEASE RETENTION.VARIANCE
RELEASE YG.SPLIT(1,*,*)
RELEASE YG.SPLIT(OFFICER.SKILL1,*,*)
END
```

```
ROUTINE TO INSTALL.BASE.PERSONNEL
  DEFINE SHARE AND EXTRA AS REAL VARIABLES
  DEFINE S,L,YG,R,C,LSIZE AS INTEGER VARIABLES
  DEFINE SUP, ELIGIBLES, INELIGIBLES AS 2-DIM INTEGER ARRAYS
  DEFINE MEMORY AS A 3-DIM INTEGER ARRAY
  RESERVE SUPPLY, OJT.ELIGIBLES, OJT.INELIGIBLES, PENDING.OJT,
          AND ROTATION.MEMORY AS N.BASE
  FOR EACH BASE DO....
     CALL DUPLICATE GIVEN YE.AUTHORIZATION(YEAR0,BASE)
                    YIELDING SUPPLY(BASE)
     CALL CONFIGURE GIVEN SUPPLY(BASE) YIELDING
        PENDING.OJT(BASE)
     CALL CONFIGURE GIVEN SUPPLY(BASE) YIELDING
        OJT.ELIGIBLES(BASE)
     CALL CONFIGURE GIVEN SUPPLY(BASE) YIELDING
       OJT.INELIGIBLES(BASE)

**INITIAL OJT ELIGIBILITY
     LET SUP(*,*)=SUPPLY(BASE)
     LET ELIGIBLES(*,*)=OJT.ELIGIBLES(BASE)
     LET INELIGIBLES(*,*)=OJT.INELIGIBLES(BASE)
     FOR EACH SKILL CALLED S
        FOR L=1 TO LEN.F(SUP(S,*)) DO.......
        LET ELIGIBLES(S,L)=.5*SUP(S,L)
        LET INELIGIBLES(S,L)=.5*SUP(S,L)
        LOOP
```

```
**INITIAL ROTATION MEMORY
     IF OVERSTAS(BASE)=1,
        LET MEMORY(*,*,*)=0
        RESERVE MEMORY(*,*,*) AS N.SKILL BY *
        FOR EACH SKILL CALLED S DO
          LET LSIZE=LEN.F(SUP(S,*))
          IF LSIZE > 0
            RESERVE MEMORY(S,*,*) AS LSIZE BY ROTATION.CYCLE+1
            LET R=1
            IF ROTATION.CYCLE=18, LET R=2 ALWAYS
            IF ROTATION.CYCLE=24, LET R=3  ALWAYS
            LET EXTRA=0
            FOR L=1 TO LSIZE, FOR C=1 TO ROTATION.CYCLE DO...
               LET SHARE=SUP(S,L)*INITIAL.ROTATION.PCT(R,C)
               ADD EXTRA TO SHARE
               LET EXTRA=FRAC.F(SHARE)
               LET MEMORY(S,L,C)=TRUNC.F(SHARE)
               LOOP
            ALWAYS
          LOOP  **TO NEXT SKILL
        REGARDLESS
     LOOP

  RELEASE INITIAL.ROTATION.PCT

**OUTPUT DATA COLLECTORS
  RESERVE ASSIGN.OUT, ASSIGN.OJT, ASSIGN.SEP, ASSIGN.SCH,
          FLOW.SEP, OJT.FLOW.OUT, TO.SCH.FLOW, FLOW.OUT,
          FLOW.IN, OJT.FLOW.IN
             AS N.BASE BY N.SKILL BY N.LEVEL

  RETURN
END
```

```
ROUTINE DYNAMIC.INITIALIZATION (HORIZON.P)

CALL INIT.PERIODS
CALL INIT.PERIOD0
CREATE EACH SCHOOL
**INITIAL SCHOOL OUTFLOWS
FOR P=1 TO MAX.LAG DO
  CALL TRAINING.PROJECTION(P)
  FOR EACH SCHOOL DO
     IF  TIME.F(P)-DURATION(SCHOOL) <= 0,
        CALL MAKE.ABLK(TECH.TRAINING.POOL(P,SCHOOL),
          SKILL.TAUGHT(SCHOOL),LEVEL.TAUGHT(SCHOOL)-1,
          BASE.LOC(SCHOOL),
          TRAINING)
          YIELDING ABLK
        LET CLASS(ABLK)=SCHOOL
        SCHEDULE A GRADUATION (ABLK,SCHOOL) AT TIME.F(P)
        ALWAYS
     LET TECH.TRAINING.POOL(P,SCHOOL)=0
     LOOP
  LOOP

LET H.MONTHS=0
CALL MAKE.ASSIGNMENTS(PERIOD0,1)
FOR P=2 TO HORIZON.P DO..
  LET H.MONTHS=P-1
  CALL MAKE.ASSIGNMENTS(P-1,P)
  LOOP
LET H.MONTHS=HORIZON.P
END
```

```
ROUTINE INIT.PERIODS
  CREATE EACH PERIOD
  SUBTRACT 1 FROM N.PERIOD
  FOR EACH PERIOD, FOR EACH BASE DO...
     LET AUTH.SUPPLY=YE.AUTHORIZATION(PLAN.YEAR(PERIOD),BASE)
     LET MIN.SUPPLY=YE.MIN.MANPOWER(PLAN.YEAR(PERIOD),BASE)
     LOOP
  FOR PERIOD=1 TO N.MONTHS+1, FOR EACH BASE DO
     CALL CONFIGURE GIVEN AUTH.SUPPLY YIELDING PROJECTION
     CALL CONFIGURE GIVEN AUTH.SUPPLY YIELDING ROTATION.POOL
     LOOP
END
```

```
ROUTINE INIT.PERIOD0
  FOR EACH BASE DO...
     LET AUTH.SUPPLY(PERIOD0,BASE)=YE.AUTHORIZATION(YEAR0,BASE)
     LET MIN.SUPPLY(PERIOD0,BASE)=YE.MIN.MANPOWER(YEAR0,BASE)
     CALL DUPLICATE GIVEN SUPPLY(BASE) YIELDING
        PROJECTION(PERIOD0,BASE)
     CALL CONFIGURE GIVEN SUPPLY(BASE) YIELDING
        ROTATION.POOL(PERIOD0,BASE)
     LOOP
END
```

```
ROUTINE TRAINING.PROJECTION (P)
  DEFINE P,S,L,SCH AS INTEGER VARIABLES
  FOR EACH SKILL CALLED S, FOR L=LEVEL3 TO LEVEL7 DO
     LET SCH=SCHOOL.CHART(S,L)
     ADD (1-OJT.PCT(S,L))*USAF.HOLES(P,S,L) TO
          TECH.TRAINING.POOL(P,SCH)
     LOOP
END
```

```
ROUTINE MAKE.ASSIGNMENTS (PMINUS1,P)
DEFINE TEMP, P, PMINUS1 AS INTEGER VARIABLES
DEFINE LEVY.FLAG, A.SUPPLY, PROJ
 AS A 2-DIMENSIONAL INTEGER ARRAY
LET PERIOD=P
RESERVE LEVY.FLAG AS N.SKILL BY N.LEVEL

**PLAN TRANSITION OF OTS AND BMT GRADS TO TECH SCHOOL
CALL LEVEL1.ASSIGNMENT(PERIOD)

**UPDATE SCHOOL PROJECTION
CALL TRAINING.PROJECTION(PERIOD)

**REMOVE OJT'S, SEPS, ROTATION, TT OUTS FROM THIS MONTH'S PROJECTION
CALL EXTRAPOLATE(PMINUS1, PERIOD)

RESERVE DEMAND AS N.BASE BY N.SKILL BY N.LEVEL
FOR EACH BASE, DO
 LET A.SUPPLY(*,*)=AUTH.SUPPLY
 LET PROJ(*,*)=PROJECTION
 FOR EACH SKILL,
   ALSO FOR LEVEL=1 TO LEN.F(A.SUPPLY(SKILL,*)), DO
     LET DEMAND(BASE,SKILL,LEVEL)=A.SUPPLY(SKILL,LEVEL)-
         PROJ(SKILL,LEVEL)
     IF DEMAND(BASE,SKILL,LEVEL) IS POSITIVE
**ALLOCATE SOME ROTATIONS TO FILL THIS DEMAND
       LET TEMP=GET.ROTATIONS(BASE,PERIOD,SKILL,LEVEL)
       SUBTRACT TEMP FROM DEMAND(BASE,SKILL,LEVEL)
       ADD TEMP TO PROJ(SKILL,LEVEL)
     REGARDLESS

     IF DEMAND(BASE,SKILL,LEVEL) IS POSITIVE AND
       LEVEL IS NOT GREATER THAN LEVEL7
**ALLOCATE SOME TECH SCHOOL GRADS TO FILL DEMAND
       LET TEMP=GET.GRADUATES(BASE,PERIOD,SKILL,LEVEL)
       SUBTRACT TEMP FROM DEMAND(BASE,SKILL,LEVEL)
       ADD TEMP TO PROJ(SKILL,LEVEL)
     REGARDLESS

     IF DEMAND(BASE,SKILL,LEVEL) IS POSITIVE
       LET LEVY.FLAG(SKILL,LEVEL)=1
     REGARDLESS
LOOP
FOR EACH SKILL,
 FOR EACH LEVEL WITH LEVY.FLAG(SKILL,LEVEL) IS EQUAL TO 1,
   CALL LEVY(PERIOD,SKILL,LEVEL)
LOOP
RELEASE DEMAND
RETURN
END **OF MAKE.ASSIGNMENT
                                                          **
```

```
ROUTINE LEVEL1.ASSIGNMENT(AP)
DEFINE AP,NEED AS INTEGER VARIABLES
FOR EACH SKILL, DO
 LET NEED=USAF.HOLES(AP,SKILL,LEVEL1)
 IF NEED IS NOT ZERO
   LET SCHOOL=SCHOOL.CHART(SKILL,LEVEL1).
   CREATE AN ABLK
   LET SIZE=NEED
   LET LVL=LEVEL1
   LET SKL=SKILL
   LET CLASS=SCHOOL
   LET PURPOSE=TRAINING
   LET DESTINATION=BASE.LOC(SCHOOL)
   IF SKILL IS NOT GREATER THAN N.AIRMEN.SKILLS
     FILE ABLK IN DISPOSITION(BMT)
   ELSE
     FILE ABLK IN DISPOSITION(OTS)
     ALWAYS
 REGARDLESS
LOOP
RETURN
END
```

```
ROUTINE TO EXTRAPOLATE(PMINUS1,P)
DEFINE PMINUS1, TO.SCH, P, TO.UPG, FROM.UPG, SEP, OJT.POOL
 AS INTEGER VARIABLES
DEFINE F AS REAL VARIABLE
DEFINE MEMORY AS A 3-DIM INTEGER ARRAY
DEFINE PROJ.OJT,ELIGIBLES,INELIGIBLES AS 2-DIM INTEGER ARRAYS
DEFINE OLD.PROJ, ROTATION, NEW.PROJ
 AS 2-DIMENSIONAL INTEGER ARRAYS
LET PERIOD=P
FOR EACH BASE, DO
 LET ROTATION(*,*)=ROTATION.POOL
 LET PROJ.OJT(*,*)=PENDING.OJT
 LET ELIGIBLES(*,*)=OJT.ELIGIBLES
 LET INELIGIBLES(*,*)=OJT.INELIGIBLES
 LET MEMORY(*,*,*)=ROTATION.MEMORY
 LET OLD.PROJ(*,*)=PROJECTION(PMINUS1,BASE)
 LET NEW.PROJ(*,*)=PROJECTION(P,BASE)
 LET TO.UPG=0
 FOR EACH SKILL CALLED S,
   ALSO FOR L=1 TO LEN.F(OLD.PROJ(S,*)), DO
     IF OVERSEAS(BASE)=1,
        LET ROTATION(S,L)=MEMORY(S,L,CELL.F(P,ROTATION.CYCLE))
        ALWAYS

     LET F=MIN.F(1,H.MONTHS/OJT.DELAY(L))
     LET OJT.POOL=ELIGIBLES(S,L)-PROJ.OJT(S,L)+F*INELIGIBLES(S,L)
     LET FROM.UPG=OJT.OUT(BASE,S,L,P,OJT.POOL)

     LET SEP=SEP.OUT(BASE,S,L,P,
        EXPECTED.SEPARATIONS(*,*,*))

     LET TO.SCH=TT.OUT(BASE,S,L,P)
     IF TO.SCH IS NOT ZERO
       CREATE AN ABLK
       LET SIZE=TO.SCH
       LET SKL=S
       LET LVL=L
       LET DESTINATION=BASE.LOC(SCHOOL.CHART(S,L))
       LET PURPOSE=RETRAINING
       LET CLASS=SCHOOL.CHART(S,L)
       FILE ABLK IN PLANNED.ASSIGNMENTS
     REGARDLESS

     LET NEW.PROJ(S,L)=
       OLD.PROJ(S,L)+TO.UPG-FROM.UPG-SEP-TO.SCH
       -ROTATION(S,L)
     ADD FROM.UPG TO PROJ.OJT(S,L)

     LET TO.UPG=FROM.UPG

 LOOP
LOOP
RETURN
END ** OF EXTRAPOLATE
```

```
ROUTINE OJT.OUT(B,S,L,P,POOL)
  DEFINE AUTH AS A 2-DIM INTEGER ARRAY
  DEFINE B,S,L,P,Y,POOL AS INTEGER VARIABLES
  DEFINE OUTS AS A REAL VARIABLE
  LET AUTH(*,*)=AUTH.SUPPLY(P,B)
  IF L=LEVEL9,
     RETURN(0)
  ELSE
     LET OUTS=OJT.PCT(S,L+1)*USAF.HOLES(P,S,L+1)
     LET OUTS=MIN.F(OUTS,POOL)
     LET Y=PLAN.YEAR(P)
     RETURN(INT.F(OUTS*AUTH(S,L)/USAF.MISSION.AUTH(Y,S,L)))
END
```

```
ROUTINE SEP.OUT (B,S,L,P,USAF.SEPS)
  DEFINE AUTH AS A 2-DIM INTEGER ARRAY
  DEFINE B,S,L,P,Y,M AS INTEGER VARIABLES
  DEFINE USAF.SEPS AS A 3-DIM REAL ARRAY
  DEFINE OUTS AS A REAL VARIABLE
   IF OVERSEAS(B) = 1 RETURN (0)
   ALWAYS
  LET AUTH(*,*)=AUTH.SUPPLY(P,B)
  LET M=CALENDAR.MONTH(P)
  LET Y=PLAN.YEAR(P)
  LET OUTS=EXIT.F(M)*USAF.SEPS(Y,S,L)
  RETURN (INT.F(OUTS*AUTH(S,L)/CONUS.USAF.M.AUTH(Y,S,L)))
END
```

```
ROUTINE TT.OUT (B,S,L,P)
  DEFINE AUTH AS A 2-DIM INTEGER ARRAY
  DEFINE B,S,L,P,Y,P2 AS INTEGER VARIABLES
  DEFINE OUTS AS A REAL VARIABLE
  LET AUTH(*,*)=AUTH.SUPPLY(P,B)
  IF L=LEVEL8 OR L=LEVEL7
     RETURN(0)
  ELSE
     LET P2=OFIL.F(P+TRAINING.TIME(S,L+1))
     LET OUTS=(1-OJT.PCT(S,L+1))*USAF.HOLES(P2,S,L+1)
     LET Y=PLAN.YEAR(P)
     RETURN(INT.F(OUTS*AUTH(S,L)/USAF.MISSION.AUTH(Y,S,L)))
END
```

```
                                                                    ..
                                                                    ..
                                                                    ..
ROUTINE TO GET.ROTATIONS(DEST,P,SK,LE)
DEFINE ROTATION AS A 2-DIM INTEGER ARRAY
DEFINE DEST, P, SZ, BS, AMOUNT, ANS, SK, LE AS INTEGER VARIABLES
LET ANS=0
LET SZ=DEMAND(DEST,SK,LE)
FOR EACH BASE CALLED BS WHILE SZ IS NOT ZERO,
     WITH BS NE DEST, DO....
 LET ROTATION(*,*)=ROTATION.POOL(P,BS)
 IF ( CONT.LOCATION(DEST)=CONUS  OR
     (CONT.LOCATION(DEST) NE CONUS AND CONT.LOCATION(BS)=CONUS))
     AND ROTATION(SK,LE) IS NOT ZERO,

     LET AMOUNT=MIN.F(SZ,ROTATION(SK,LE))
     SUBTRACT AMOUNT FROM SZ
     SUBTRACT AMOUNT FROM ROTATION(SK,LE)
     CREATE AN ABLK
     LET SIZE=AMOUNT
     LET SKL=SK
     LET LVL=LE
     LET DESTINATION=DEST
     LET PURPOSE=ROTATING
     FILE ABLK IN PLANNED.ASSIGNMENTS(P,BS)
     ADD AMOUNT TO ANS
     REGARDLESS
 LOOP

RETURN WITH ANS
END ''OF GET.ROTATIONS
                                                                    ..
```

```
ROUTINE TO GET.GRADUATES(DEST,HP,SK,LE)
DEFINE DEST, SZ,  LE, P, HP, SK, AMOUNT, ANS, SCH
 AS INTEGER VARIABLES
LET ANS=0
LET SZ=DEMAND(DEST,SK,LE)
LET SCH=SCHOOL.CHART(SK,LE)
FOR P=PERIOD.F(TIME.V) TO HP, WHILE SZ IS NOT ZERO, DO....
 IF TECH.TRAINING.POOL(P,SCH) IS NOT ZERO
   LET AMOUNT=MIN.F(SZ,TECH.TRAINING.POOL(P,SCH))
   SUBTRACT AMOUNT FROM SZ
   SUBTRACT AMOUNT FROM TECH.TRAINING.POOL(P,SCH)
   CREATE AN ABLK
   LET SIZE=AMOUNT
   LET SKL=SK
   LET LVL=LE
   LET DESTINATION=DEST
   LET DEPARTURE.DATE=TIME.F(P)
   LET PURPOSE=ASSIGN
   FILE THIS ABLK IN DISPOSITION(SCH)
   ADD AMOUNT TO ANS
   ALWAYS
LOOP
RETURN WITH ANS
END ** OF GET.GRADUATE
```

```
ROUTINE LEVY(P,SK,LE)
DEFINE POS, BP1, POOL, FLAG, P, NEG, BP, BN AS INTEGER VARIABLES
DEFINE PROJ, PROJ1, M.SUPPLY AS 2-DIMENSIONAL ARRAYS
DEFINE RATE AS REAL VARIABLE
LET FLAG=0
'RESTART'
LET POS=0
LET NEG=0
FOR EACH BASE,DO
 IF DEMAND(BASE,SK,LE) IS POSITIVE
   ADD DEMAND(BASE,SK,LE) TO POS
 ELSE
 SUBTRACT DEMAND(BASE,SK,LE) FROM NEG
ALWAYS LOOP
IF POS IS NOT GREATER THAN NEG
 LET RATE=POS/NEG
 GO TO 'COMP'
ELSE
IF FLAG IS EQUAL TO 1  LET RATE=1 GO TO 'COMP'
ELSE
FOR EACH BASE, DO
 LET PROJ(*,*)=PROJECTION(P,BASE)
 LET M.SUPPLY(*,*)=MIN.SUPPLY(P,BASE)
 LET DEMAND(BASE,SK,LE)=M.SUPPLY(SK,LE)-PROJ(SK,LE)
LOOP
LET FLAG=1
GO TO 'RESTART'
'COMP'
LET BP1=1
FOR EACH BASE CALLED BN WITH DEMAND(BN,SK,LE) IS NEGATIVE, DO
 LET PROJ1(*,*)=PROJECTION(P,BN)
 LET POOL=-DEMAND(BN,SK,LE)*RATE
 LET BP=BP1
 ALSO FOR BP1=BP TO N.BASE WITH DEMAND(BP1,SK,LE) IS POSITIVE
   WHILE POOL IS NOT ZERO, DO
     CREATE AN ABLK
     LET SIZE=MIN.F(POOL,DEMAND(BP1,SK,LE))
     LET SKL=SK
     LET LVL=LE
     LET DESTINATION=BP1
     LET PURPOSE=LEVYING
     FILE ABLK IN PLANNED.ASSIGNMENTS(P,BN)
     SUBTRACT SIZE FROM POOL
     SUBTRACT SIZE FROM DEMAND(BP1,SK,LE)
     ADD SIZE TO DEMAND(BN,SK,LE)
     SUBTRACT SIZE FROM PROJ1(SK,LE)
     LET PROJ(*,*)=PROJECTION(P,BP1)
     ADD SIZE TO PROJ(SK,LE)
LOOP
RETURN
END '' OF LEVY
```

```
ROUTINE FIND.ASSIGNMENT(S,L,BA,REASON,AMOUNT)
**
DEFINE S,L,BA,AMOUNT,AMOUNT.LEFT,BS,TOTAL,SHARE
       AS INTEGER VARIABLES
**
SUBSTITUTE THESE 4 LINES FOR FOR.ALL
FOR EACH BASE CALLED BS WITH BS NOT EQUAL TO BA
AND OVERSEAS(BA)+OVERSEAS(BS) NOT EQUAL TO 2, DO
   LET SUPP(*,*)=SUPPLY(BS)
   IF LEN.F(SUPP(S,*)) IS GREATER THAN L
SUBSTITUTE THESE 8 LINES FOR PLACE.BLK
CREATE AN ABLK
LET SIZE=SHARE
LET SKL=S
LET LVL=L
LET PURPOSE=ASSIGN
LET DESTINATION=BS
LET DEPARTURE.DATE=TIME.V
CALL CHECK.PIPE(ABLK,PIPE.CHART(BA,BS))
DEFINE SUPP AS A 2-DIMENSIONAL INTEGER ARRAY
LET TOTAL=0
FOR.ALL
   ADD SUPP(S,L)TO TOTAL
   ALWAYS LOOP
LET AMOUNT.LEFT=AMOUNT
FOR.ALL
   LET SHARE=AMOUNT*SUPP(S,L)/TOTAL
   IF SHARE IS GREATER THAN 0
   PLACE.BLK
   SUBTRACT SIZE FROM AMOUNT.LEFT
   ALWAYS ALWAYS LOOP
IF REASON IS EQUAL TO ROTATING
LET SUPP(*,*)=SUPPLY(BA)
SUBTRACT AMOUNT FROM SUPP(S,L)
REGARDLESS
IF AMOUNT.LEFT IS ZERO
RETURN
ALWAYS
FOR.ALL
   GO TO 'OVER'
   ALWAYS LOOP
STOP
'OVER'
LET SHARE=AMOUNT.LEFT
PLACE.BLK
RETURN
END
```

```
ROUTINE FIND.SCHOOL(OLD.SCHOOL,AMOUNT) ''FOR EXCESS OTS,BMT GRADS
DEFINE OLD.SCHOOL,NEW.SCHOOL,AMOUNT AS INTEGER VARIABLES
IF OLD.SCHOOL = BMT
   LET NEW.SCHOOL=RANDI.F(1,3*N.AIRMEN.SKILLS,1)
   ELSE
   LET NEW.SCHOOL=RANDI.F(3*N.AIRMEN.SKILLS+1,N.TECH.SCHOOLS,1)
ALWAYS
CREATE AN ABLK
LET SIZE=AMOUNT
LET SKILL=SKILL.TAUGHT(NEW.SCHOOL)
LET LEVEL=HELPER
LET DESTINATION=BASE.LOC(NEW.SCHOOL)
LET CLASS=NEW.SCHOOL
LET PURPOSE=TRAINING
LET DEPARTURE.DATE=TIME.V
CALL CHECK.PIPE(ABLK,PIPE.CHART(BASE.LOC(OLD.SCHOOL),DESTINATION))
END
```

```
                                                                      ''
ROUTINE TO CHECK.SCHOOL(BLK,SCH)
DEFINE BLK, NEW.BLK, SCH AS INTEGER VARIABLES
LET SCHOOL=SCH
IF ENROLLMENT IS EQUAL TO SCHOOL.CAPACITY
 FILE BLK LAST IN SCHOOL.QUEUE
 ADD SIZE(BLK) TO SCHOOL.QUEUE.SIZE
 RETURN
ELSE
IF SIZE(BLK) IS GREATER THAN SCHOOL.CAPACITY-ENROLLMENT
  PERFORM SPLIT(BLK,SCHOOL.CAPACITY-ENROLLMENT) YIELDING NEW.BLK
 FILE NEW.BLK   IN SCHOOL.QUEUE
 ADD SIZE(NEW.BLK) TO SCHOOL.QUEUE.SIZE
REGARDLESS
ADD SIZE(BLK) TO ENROLLMENT
SCHEDULE A GRADUATION(BLK,SCHOOL) IN DURATION UNITS
END '' OF CHECK.SCHOOL
                                                                      ''
```

```
                                                                    ..
ROUTINE TO CHECK.TRAVELPIPE(BLK,PIPELINE)
DEFINE BLK, NEW.BLK, PIPELINE AS INTEGER VARIABLES
IF VOLUME(PIPELINE) IS EQUAL TO CAPACITY(PIPELINE)
 FILE BLK LAST IN WAITING.QUEUE(PIPELINE)
 ADD SIZE(BLK) TO WAIT.QUEUE.SIZE
 RETURN
ELSE
IF SIZE(BLK) IS GREATER THAN CAPACITY(PIPELINE)-VOLUME(PIPELINE)
 PERFORM SPLIT(BLK,CAPACITY(PIPELINE)-VOLUME(PIPELINE)) YIELDING NEW.BLK
 FILE NEW.BLK   IN WAITING.QUEUE(PIPELINE)
 ADD SIZE(BLK) TO WAIT.QUEUE.SIZE
REGARDLESS
ADD SIZE(BLK) TO VOLUME(PIPELINE)
SCHEDULE A PIPE.EXIT(BLK,PIPELINE) IN MIN.TRAVEL.TIME(PIPELINE) UNITS
RETURN
END ..OF CHECK.TRAVELPIPE
                                                                    ..
```

```
ROUTINE ASSIGN.CHECK GIVEN BLK,PERIOD,BASE
  DEFINE BLK,PERIOD,BASE AS INTEGER VARIABLES
GO TO PUR(PURPOSE(BLK))
'PUR(ASSIGN)' 'PUR(ROTATING)' 'PUR(LEVYING)'
  ADD SIZE(BLK) TO ASSIGN.OUT(BASE,SKL(BLK),LVL(BLK))
  RETURN
'PUR(OJT.UPGRADE)'
  ADD SIZE(BLK) TO ASSIGN.OJT(BASE,SKL(BLK),LVL(BLK))
  RETURN
'PUR(SEPARATING)'
  ADD SIZE(BLK) TO ASSIGN.SEP(BASE,SKL(BLK),LVL(BLK))
  RETURN
'PUR(TRAINING)' 'PUR(RETRAINING)'
  ADD SIZE(BLK) TO ASSIGN.SCH(BASE,SKL(BLK),LVL(BLK))
  RETURN
END
```

```
ROUTINE CREDIT(N,S,L,B,REASON)
  DEFINE N,S,L,B,REASON AS INTEGER VARIABLES
  DEFINE MEMORY AS A 3-DIM INTEGER ARRAY
  DEFINE SUP,ELIGIBLES, INELIGIBLES AS 2-DIM INTEGER ARRAYS
  DEFINE SHARE AND EXTRA AS A REAL VARIABLE
  DEFINE THIS.P,P,CYCLE,LIM AS INTEGER VARIABLES
  IF N=0, RETURN ELSE
  LET SUP(*,*)=SUPPLY(B)
  LET ELIGIBLES(*,*)=OJT.ELIGIBLES(B)
  LET INELIGIBLES(*,*)=OJT.INELIGIBLES(B)

  ADD N TO SUP(S,L)

  IF OVERSEAS(B)=1
     LET MEMORY(*,*,*)=ROTATION.MEMORY(B)
     LET THIS.P=PERIOD.F(TIME.V)
     LET CYCLE=ROTATION.CYCLE(B)
     IF REASON NE OJT.UPGRADE
        ADD N TO MEMORY(S,L,CELL.F(THIS.P,CYCLE))
     ELSE
        LET SHARE=N/CYCLE
        LET LIM=P+CYCLE
        FOR P=THIS.P TO LIM DO
           LET EXTRA=FRAC.F(SHARE)
           ADD TRUNC.F(SHARE) TO MEMORY(S,L,CELL.F(P,CYCLE))
           ADD EXTRA TO SHARE
           LOOP
        ALWAYS
     ALWAYS

  IF (REASON=OJT.UPGRADE OR REASON=ASSIGN) AND L < LEVEL7
     ADD N TO INELIGIBLES(S,L)
  ELSE
     ADD N TO ELIGIBLES(S,L)
   ALWAYS
END
```

```
ROUTINE DEBIT (N,S,L,B,REASON)
  DEFINE N,S,L,B,REASON,THIS.P,P,CYCLE,LIM AS INTEGER
     VARIABLES
  DEFINE MEMORY AS A 3-DIM INTEGER ARRAY
  DEFINE SUP,ELIGIBLES, INELIGIBLES AS 2-DIM INTEGER ARRAYS
  DEFINE SHARE, EXTRA AS REAL VARIABLES
  IF N=0, RETURN ELSE
  LET SUP(*,*)=SUPPLY(B)
  LET ELIGIBLES(*,*)=OJT.ELIGIBLES(B)
  LET INELIGIBLES(*,*)=OJT.INELIGIBLES(B)

  SUBTRACT N FROM SUP(S,L)

  IF OVERSEAS(B)=1,
     LET CYCLE=ROTATION.CYCLE(B)
     LET MEMORY(*,*,*)=ROTATION.MEMORY(B)
     LET THIS.P=PERIOD.F(TIME.V)
     IF REASON =ROTATING,
        SUBTRACT N FROM MEMORY(S,L,CELL.F(THIS.P,CYCLE))
     ELSE
         LET SHARE=N/CYCLE
        LET LIM=P+CYCLE
        FOR P=THIS.P TO LIM DO
           LET EXTRA=FRAC.F(SHARE)
           SUBTRACT TRUNC.F(SHARE) FROM MEMORY(S,L,CELL.F(P,CYCLE))
           ADD EXTRA TO SHARE
           LOOP
        ALWAYS
     ALWAYS

  IF REASON=RETRAINING
     SUBTRACT N FROM INELIGIBLES(S,L)
     IF INELIGIBLES(S,L) < 0
        SUBTRACT -INELIGIBLES(S,L) FROM ELIGIBLES(S,L)
        LET INELIGIBLES(S,L)=0
        ALWAYS
  ELSE
     SUBTRACT N FROM ELIGIBLES(S,L)
     IF ELIGIBLES(S,L) < 0,
       SUBTRACT - ELIGIBLES(S,L) FROM INELIGIBLES(S,L)
       LET ELIGIBLES(S,L)=0
       ALWAYS
  ALWAYS
  RETURN
END
```

```
                                                                    ''
                                                                    ''
                                                                    ''
                                                                    ''
                                                                    ''
EVENT ASSIGNMENT
DEFINE HORIZON.PERIOD AS AN INTEGER VARIABLE
  LET HORIZON.PERIOD=PERIOD.F(TIME.V+H.MONTHS)
  CALL MAKE.ASSIGNMENTS(HORIZON.PERIOD-1,HORIZON.PERIOD)
  IF HORIZON.PERIOD < N.PERIOD  ''CONTINUE ASSIGNMENT PLANNING
     SCHEDULE AN ASSIGNMENT IN 1 UNIT
     REGARDLESS
END
```

```
                                                                    ;;
                                                                    ;;
                                                                    ;;
EVENT GRADUATION(GG.BLK,GG.SCH)
  DEFINE GG.BLK,GG.SCH AS INTEGER VARIABLES
DEFINE NEW.ORDER, BLK, ORDER,NEW.BLK AS INTEGER VARIABLES
  CALL  FLOW7(GG.BLK,GG.SCH)
LET ABLK=GG.BLK
LET SCHOOL=GG.SCH
WHILE DISPOSITION IS NOT EMPTY AND SIZE IS NOT ZERO, DO
 REMOVE FIRST ORDER FROM DISPOSITION
 IF SIZE(ORDER) IS GREATER THAN SIZE
    PERFORM SPLIT (ORDER,SIZE) YIELDING NEW.ORDER
   FILE NEW.ORDER FIRST IN DISPOSITION
 REGARDLESS
 PERFORM CHECK.TRAVELPIPE(ORDER,PIPE.CHART(BASE.LOC,DESTINATION(ORDER)))
 SUBTRACT SIZE(ORDER) FROM SIZE
LOOP
IF LVL=LEVEL1
  CALL FIND.SCHOOL(SCHOOL,SIZE)
ELSE
IF SIZE IS NOT ZERO
 CALL FIND.ASSIGNMENT(SKL,LVL+1,BASE.LOC,TRAINING,SIZE)
REGARDLESS ALWAYS
DESTROY THIS ABLK
WHILE SCHOOL.QUEUE IS NOT EMPTY AND ENROLLMENT IS
 LESS THAN SCHOOL.CAPACITY, DO
   LET BLK=F.SCH.Q
   IF SIZE(BLK)> SCHOOL.CAPACITY-ENROLLMENT
     CALL SPLIT GIVEN BLK,(SIZE(BLK)+ENROLLMENT-SCHOOL.CAPACITY)
                YIELDING NEW.BLK
   ELSE REMOVE FIRST NEW.BLK FROM SCHOOL.QUEUE
   ALWAYS SUBTRACT SIZE(NEW.BLK) FROM SCHOOL.QUEUE.SIZE
   PERFORM CHECK.SCHOOL (NEW.BLK,SCHOOL)
   LOOP
RETURN
END ''OF GRADUATION
                                                                    ;;
```

```
;;
;;
;;
;;
EVENT INDUCTION SAVING THE EVENT NOTICE
CREATE AN ABLK
LET SIZE=OFFICER.RECRUITS(PERIOD.F(TIME.V))
LET LVL=0
LET PURPOSE=TRAINING
LET CLASS=OTS
PERFORM CHECK.SCHOOL(ABLK,OTS)
CREATE AN ABLK
LET SIZE=AIRMEN.RECRUITS(PERIOD.F(TIME.V))
LET LVL=0
LET PURPOSE=TRAINING
LET CLASS=BMT
PERFORM CHECK.SCHOOL(ABLK,BMT)
SCHEDULE THIS INDUCTION IN 1 UNIT
RETURN
END **OF INDUCTION
                                                    ;;
```

```
EVENT MASSIVE.RIF
  RELEASE ALL.THE.OFFICERS AND ALL.THE.AIRMEN
  DESTROY THE AIRFORCE
  GO HOME


'HOME'
  STOP
''FINISH
END  ''OF AIRFORCE
```

```
EVENT MOVEMENTS SAVING THE EVENT NOTICE
DEFINE SUP, ROT.PL AS A 2-DIMENSIONAL INTEGER ARRAYS
DEFINE GROUP,S,L,P AS INTEGER VARIABLES
  DEFINE ELIGIBLES,INELIGIBLES,PROJ.OJT AS 2-DIM INTEGER ARRAYS
LET PERIOD=PERIOD.F(TIME.V)
LET P=PERIOD
FOR EACH BASE, DO
 LET PROJ.OJT(*,*)=PENDING.OJT
 LET ELIGIBLES(*,*)=OJT.ELIGIBLES
 LET INELIGIBLES(*,*)=OJT.INELIGIBLES
 LET SUP(*,*)=SUPPLY
 LET ROT.PL(*,*)=ROTATION.POOL
 FOR EACH SKILL CALLED S
     FOR L=1 TO LEN.F(SUP(S,*)) DO.....
**SEPARATIONS
     LET GROUP=SEP.OUT(BASE,S,L,P,
             ACTUAL.SEPARATIONS(*,*,*))
     CALL DEBIT(GROUP,S,L,BASE,SEPARATING)
     IF SEP.POINT(BASE) NE BASE,
        CALL MAKE.ABLK(GROUP,S,L,SEP.POINT(BASE),SEPARATING)
           YIELDING ABLK
       PERFORM CHECK.PIPELINE(ABLK,PIPE.CHART(BASE,SEP.POINT(BASE))
       REGARDLESS

**OJT UPGRADES
     LET GROUP=INELIGIBLES(S,L)/OJT.DELAY(L)
     SUBTRACT GROUP FROM INELIGIBLES (S,L)
     ADD GROUP TO ELIGIBLES(S,L)

     LET GROUP=OJT.OUT(BASE,S,L,P,ELIGIBLES(S,L))
     CALL DEBIT (GROUP,S,L,BASE,OJT.UPGRADE)
     CALL CREDIT (GROUP,S,L+1,BASE,OJT.UPGRADE)
     SUBTRACT GROUP FROM PROJ.OJT(S,L)
     LOOP
```

```
**ROTATION, TECH TRAINING  SCHOOL, LEVY ASSIGNMENTS
 WHILE PLANNED.ASSIGNMENTS IS NOT EMPTY, DO
   REMOVE FIRST ABLK FROM PLANNED.ASSIGNMENTS
   LET DEPARTURE.DATE=TIME.V
   GO TO PUR(PURPOSE)
*PUR(LEVYING)*
*PUR(ROTATING)*
*PUR(ASSIGN)*
*PUR(RETRAINING)*
   CALL DEBIT(SIZE(ABLK),SKL(ABLK),LVL(ABLK),BASE,PURPOSE)
*PUR(TRAINING)*
   PERFORM CHECK.TRAVELPIPE(ABLK,PIPE.CHART(BASE,DESTINATION))
  LOOP

 FOR EACH SKILL,
   ALSO FOR LEVEL=1 TO LEN.F(ROT.PL(SKILL,*)), DO
     IF ROT.PL(SKILL,LEVEL)>0
       CALL FIND.ASSIGNMENT(SKILL,LEVEL,BASE,ROTATING,ROT.PL(SKILL,
          LEVEL))
       ALWAYS
      LOOP
 LOOP
SCHEDULE THIS MOVEMENTS IN 1 UNIT
RETURN
END ** OF EVENT MOVEMENTS
```

```
EVENT PERSONNEL.SAVING SAVING THE EVENT NOTICE
  DEFINE SUPP AS A 2-DIM INTEGER ARRAY
USE UNIT 2 FOR OUTPUT
WRITE CALENDAR.MONTH(PERIOD.F(TIME.V)) AS "MONTH ",I 3
FOR EACH BASE, DO
  LET SUPP(*,*)=SUPPLY
  ALSO FOR EACH SKILL,DO
     ALSO FOR EACH LEVEL, DO
     IF LEN.F(SUPP(SKILL,*))<LEVEL
        WRITE 0,0,0,0,0,0,0 AS I 4,6 I 3
     ELSE
        WRITE SUPP(SKILL,LEVEL),FLOW.SEP(SKILL,LEVEL),
              OJT.FLOW.OUT(SKILL,LEVEL),TO.SCH.FLOW(SKILL,LEVEL),
              FLOW.OUT(SKILL,LEVEL),FLOW.IN(SKILL,LEVEL),OJT.FLOW.IN AS
              I 4,6 I 3
  ALWAYS LOOP
WRITE AS "SCHOOL "
FOR EACH SCHOOL, DO
  WRITE ENROLLMENT(SCHOOL),TOTAL.NEW.STUDENTS(SCHOOL),
        TOTAL.GRADUATION(SCHOOL),TOTAL.SCH.QUEUE(SCHOOL),
        PEAK.SCH.QUEUE(SCHOOL) AS 5 I 5
  LOOP
WRITE AS "PIPE  "
FOR EACH TRAVELPIPE, DO
  WRITE VOLUME(TRAVELPIPE),TRAVEL.VOLUME(TRAVELPIPE),
        PEAK.VOLUME(TRAVELPIPE),TOTAL.WAIT.QUEUE(TRAVELPIPE),
        PEAK.WAIT.QUEUE(TRAVELPIPE) AS 5 I 6
     LOOP
FOR EACH BASE,
  FOR EACH SKILL,
     FOR EACH LEVEL, DO
        LET FLOW.SEP=0
        LET OJT.FLOW.OUT=0
        LET TO.SCH.FLOW=0
        LET FLOW.OUT=0
        LET FLOW.IN=0
  LOOP
FOR EACH SCHOOL, DO
  LET TOTAL.NEW.STUDENTS=0
  LET TOTAL.GRADUATION=0
  LET TOTAL.SCH.QUEUE=0
  LET PEAK.SCH.QUEUE=0
  LOOP
FOR EACH TRAVELPIPE, DO
  LET TRAVEL.VOLUME=0
  LET PEAK.VOLUME=0
  LET TOTAL.WAIT.QUEUE=0
  LET PEAK.WAIT.QUEUE=0
  LOOP
SCHEDULE THIS PERSONNEL.SAVING IN 1 UNIT
RETURN
END
```

```
EVENT PIPE.EXIT(P.BLK,PIPELINE)
DEFINE BLK,NEW.BLK AS INTEGER VARIABLES
 DEFINE PIPELINE AS AN INTEGER VARIABLE
LET ABLK=P.BLK
LET TRAVELPIPE=PIPELINE
SUBTRACT SIZE FROM VOLUME
IF DESTINATION IS NOT EQUAL TO EXIT.BASE
 PERFORM CHECK.TRAVELPIPE(ABLK,PIPE.CHART(EXIT.BASE,DESTINATION))
 GO TO 'CHECK.QUEUE'
ELSE
GO TO 'EXIT(PURPOSE)'
'EXIT(ASSIGN)'
CALL CREDIT (SIZE,SKL,LVL,DESTINATION,ASSIGN)
DESTROY THIS ABLK
GO TO 'CHECK.QUEUE'
'EXIT(TRAINING)'
'EXIT(RETRAINING)'
PERFORM CHECK.SCHOOL(ABLK,CLASS)
GO TO 'CHECK.QUEUE'
'EXIT(SEPARATING)'
DESTROY THIS ABLK
'CHECK.QUEUE'
WHILE WAITING.QUEUE IS NOT EMPTY AND VOLUME IS LESS THAN CAPACITY, DO
  LET BLK=F.WAITING.QUEUE
  IF SIZE(BLK)>CAPACITY-VOLUME
     CALL SPLIT GIVEN BLK,(SIZE(BLK)+VOLUME-CAPACITY)
                  YIELDING NEW.BLK
  ELSE REMOVE FIRST NEW.BLK FROM WAITING.QUEUE
     ALWAYS SUBTRACT SIZE(NEW.BLK) FROM WAIT.QUEUE.SIZE
  PERFORM CHECK.TRAVELPIPE (NEW.BLK,TRAVELPIPE)
  LOOP
RETURN
END ''OF PIPE.EXIT
```

```
EVENT SAVE.ASSIGN SAVING THE EVENT NOTICE
  DEFINE SUPP AS A 2-DIM INTEGER ARRAY
  DEFINE Z AS AN INTEGER VARIABLE
LET ASMT.PER=CALENDAR.MONTH(PERIOD.F(TIME.V)+4.MONTHS)
LET Z=0
USE UNIT 1 FOR OUTPUT
WRITE ASMT.PER AS "MONTH ",I 3
FOR EACH BASE, DO
  LET SUPP(*,*)=PROJECTION
  ALSO FOR EACH SKILL, DO
     ALSO FOR EACH LEVEL, DO
        IF LEN.F(SUPP(SKILL,*))<LEVEL
           WRITE Z,Z,Z,Z,Z AS I 4,4 I 3
           GO TO 'LOOP'
        ELSE
           WRITE SUPP(SKILL,LEVEL),ASSIGN.SEP(SKILL,LEVEL),
                                   ASSIGN.OJT(SKILL,LEVEL),
                                   ASSIGN.SCH(SKILL,LEVEL),
                                   ASSIGN.OUT(SKILL,LEVEL)
                                   AS I 4,4 I 3
  'LOOP' LOOP
FOR EACH SCHOOL, DO
  LET SKILL=SKILL.TAUGHT
  LET LEVEL=LEVEL.ENTER
   WRITE TECH.TRAINING.POOL(ASMT.PER,SCHOOL),
         ENTER.SCHOOL.TABLE(ASMT.PER,SCHOOL) AS 2 I 5
  LOOP
FOR EACH BASE
  FOR EACH SKILL,
     FOR EACH LEVEL, DO
        LET ASSIGN.OUT=0
        LET ASSIGN.OJT=0
        LET ASSIGN.SEP=0
        LET ASSIGN.SCH=0
  LOOP
SCHEDULE THIS SAVE.ASSIGN IN 1 UNIT
RETURN
END
```

```
ROUTINE FLOW1 GIVEN BLK,P,B
  DEFINE BLK,P AND B AS INTEGER VARIABLES
GO TO 'PUR(PURPOSE(BLK))'
'PUR(SEPARATING)'
  ADD SIZE(BLK) TO FLOW.SEP(B,SKL(BLK),LVL(BLK))
  RETURN
'PUR(OJT.UPGRADE)'
  ADD SIZE(BLK) TO OJT.FLOW.OUT(B,SKL(BLK),LVL(BLK))
  RETURN
'PUR(TRAINING)' 'PUR(RETRAINING)'
  ADD SIZE(BLK) TO TO.SCH.FLOW(B,SKL(BLK),LVL(BLK))
  ADD SIZE(BLK) TO ENTER.SCHOOL.TABLE(P,SCHOOL.CHART
                  (SKL(BLK),LVL(BLK)))
  RETURN
'PUR(ROTATING)' 'PUR(LEVYING)' 'PUR(ASSIGN)'
  ADD SIZE(BLK) TO FLOW.OUT(B,SKL(BLK),LVL(BLK))
  RETURN
END


ROUTINE FLOW2 GIVEN NOTICE AND TIME
  DEFINE NOTICE AS INTEGER VARIABLE
  DEFINE TIME AS REAL VARIABLE
ADD SIZE(G.BLK(NOTICE)) TO TOTAL.NEW.STUDENTS(G.SCH(NOTICE))
LET PEAK.ENROLLMENT(G.SCH(NOTICE))=MAX.F(ENROLLMENT(G.SCH(NOTICE))+
    SIZE(G.BLK(NOTICE)),PEAK.ENROLLMENT(G.SCH(NOTICE)))
RETURN
END


ROUTINE FLOW3 GIVEN BLK
  DEFINE BLK AS INTEGER VARIABLE
GO TO 'PUR(PURPOSE(BLK))'
'PUR(OJT.UPGRADE)'
  ADD SIZE(BLK) TO OJT.FLOW.IN(DESTINATION(BLK),SKL(BLK),LVL(BLK)+1)
  RETURN
'PUR(ASSIGN)'
  ADD SIZE(BLK) TO FLOW.IN(DESTINATION(BLK),SKL(BLK),LVL(BLK))
  RETURN
'PUR(RETRAINING)' 'PUR(TRAINING)' 'PUR(SEPARATING)'
  RETURN
END


ROUTINE FLOW4 GIVEN BLK,SCH
  DEFINE BLK AND SCH AS INTEGER VARIABLES
ADD SIZE(BLK) TO TOTAL.SCH.QUEUE(SCH)
LET PEAK.SCH.QUEUE(SCH)=MAX.F(PEAK.SCH.QUEUE(SCH),
   SCHOOL.QUEUE.SIZE(SCH)+SIZE(BLK))
RETURN
END
```

```
ROUTINE FLOW5 GIVEN NOTICE AND TIME
  DEFINE NOTICE AS INTEGER VARIABLE
  DEFINE TIME AS REAL VARIABLE
ADD SIZE(P.BLK(NOTICE)) TO TRAVEL.VOLUME(PIPE(NOTICE))
LET PEAK.VOLUME(PIPE(NOTICE))=MAX.F(VOLUME(PIPE(NOTICE))+
    SIZE(P.BLK(NOTICE)),PEAK.VOLUME(PIPE(NOTICE)))
RETURN
END


ROUTINE FLOW6 GIVEN BLK AND PIPELINE
  DEFINE BLK AND PIPELINE AS INTEGER VARIABLES
ADD SIZE(BLK) TO TOTAL.WAIT.QUEUE(PIPELINE)
LET PEAK.WAIT.QUEUE(PIPELINE)=MAX.F(PEAK.WAIT.QUEUE(PIPELINE),
    WAIT.QUEUE.SIZE(PIPELINE)+SIZE(BLK))
RETURN
END


ROUTINE FLOW7 GIVEN BLK AND SCH
  DEFINE BLK AND SCH AS INTEGER VARIABLES
ADD SIZE(BLK) TO TOTAL.GRADUATION(SCH)
RETURN
END
```

```
ROUTINE CALENDAR.MONTH (P)
  DEFINE P AS AN INTEGER VARIABLE
  RETURN WITH MOD.F(P-1,12)+1
END



ROUTINE CEIL.F (R)
  DEFINE R,F AS REAL VARIABLES
LET F=FRAC.F(R)
IF F=0 RETURN WITH TRUNC.F(R)
ELSE RETURN WITH TRUNC.F(R)+1
END



,,
ROUTINE CELL.F(P,CYCLE)
  DEFINE P,CYCLE AS INTEGER VARIABLES
  RETURN WITH MOD.F(P-1,CYCLE+1)+1
END



ROUTINE CONFIGURE GIVEN A YIELDING B
  DEFINE A AND B AS 2-DIM INTEGER ARRAYS
  DEFINE I,ISIZE,JSIZE AS INTEGER VARIABLES
LET ISIZE=DIM.F(A(*,*))
RESERVE B(*,*) AS ISIZE BY *
FOR I=1 TO ISIZE DO
  LET JSIZE=LEN.F(A(I,*))
  IF JSIZE>0
     RESERVE B(I,*) AS JSIZE
     ALWAYS
  LOOP
END
```

```
ROUTINE DUPLICATE GIVEN A YIELDING B
  DEFINE A AND B AS 2-DIM INTEGER ARRAYS
  DEFINE I,J,ISIZE,JSIZE AS INTEGER VARIABLES
LET ISIZE=DIM.F(A(*,*))
RESERVE B(*,*) AS ISIZE BY *
FOR I=1 TO ISIZE DO
  LET JSIZE=LEN.F(A(I,*))
  IF JSIZE>0
     RESERVE B(I,*) AS JSIZE
     FOR J=1 TO JSIZE
        STORE A(I,J) IN B(I,J)
     ALWAYS
  LOOP
END
```

```
ROUTINE LEN.F(ROW)
DEFINE ROW AS A 1-DIM INTEGER ARRAY
IF ROW(*) EQ 0
   RETURN(0)
ELSE
  RETURN(DIM.F(ROW(*)))
END
```

```
ROUTINE MAKE.ABLK GIVEN N,S,L,D,USAGE YIELDING NEW.BLK
  DEFINE N,S,L,D,USAGE,NEW.BLK AS INTEGER VARIABLES
CREATE AN ABLK
  LET SIZE=N
  LET SKL=S
  LET LVL=L
  LET PURPOSE=USAGE
  LET NEW.BLK=ABLK
END
```

```
ROUTINE PERIOD.F (T)
  DEFINE T AS A REAL VARIABLE
  RETURN WITH TRUNC.F(T)+1
END
```

```
ROUTINE PLAN.YEAR (P)
  DEFINE P AS AN INTEGER VARIABLE
  RETURN WITH DIV.F (P-1,12)+1
END
```

```
ROUTINE FOR READCHECK GIVEN TEST
DEFINE CHECK,TEST AS ALPHA VARIABLES
IF CARD IS NOT NEW, START NEW CARD
ALWAYS
READ CHECK
WRITE CHECK AS /,"GOT TO  ",A 10,/
IF CHECK = TEST
   RETURN
ELSE
   WRITE TEST,CHECK AS /," INPUT ERROR : EXPECTED ",A 10,
         " READ ",A 10," RUN ABORTED..."
   STOP
END
```

```
ROUTINE SPLIT (OLD.BLK,AMOUNT) YIELDING NEW.BLK
DEFINE NEW.BLK,OLD.BLK AND AMOUNT AS INTEGER VARIABLES
CREATE AN ABLK CALLED NEW.BLK
LET SKL(NEW.BLK)=SKL(OLD.BLK)
LET LVL(NEW.BLK)=LVL(OLD.BLK)
LET DESTINATION(NEW.BLK)=DESTINATION(OLD.BLK)
LET DEPARTURE.DATE(NEW.BLK)=DEPARTURE.DATE(OLD.BLK)
LET PURPOSE(NEW.BLK)=PURPOSE(OLD.BLK)
LET CLASS(NEW.BLK)=CLASS(OLD.BLK)
LET SIZE(NEW.BLK)=SIZE(OLD.BLK)-AMOUNT
LET SIZE(OLD.BLK)=AMOUNT
RETURN
END ** OF SPLIT
```

```
ROUTINE TIME.F (P)
  DEFINE P AS AN INTEGER VARIABLE
  RETURN WITH REAL.F(P-1)
END
```

```
''
''DEBUG ROUTINES...
  ROUTINE TO PR.YG.ARRAY GIVEN Y
  DEFINE Y AS AN INTEGER VARIABLE
''
WRITE " YG.ARRAY" AS /,A 10
LET YG.ARRAY(*,*,*)=YEAR.GROUPS(Y)
  FOR EACH SKILL CALLED S,    FOR EACH LEVEL CALLED L, DO
     WRITE Y,S,L AS /,3 I 3
     FOR YG=YG1 TO YG30 DO
        WRITE YG.ARRAY(S,L,YG) AS  I 4
        LOOP
     LOOP
RETURN
END
```

```
''
  ROUTINE TO PR.USAF.PROJ GIVEN Y AND ARRAY
  DEFINE Y AS AN INTEGER VARIABLE
  DEFINE ARRAY AS A 2-DIM REAL ARRAY
''
  WRITE " USAF.PROJ" AS /,A 10
  FOR EACH SKILL CALLED S, DO
     WRITE Y,S AS /,2 I 3
     FOR EACH LEVEL CALLED L, DO
        WRITE ARRAY(S,L) AS  D(8,2)
        LOOP
     LOOP
  RETURN
  END
```

```
EVENT STORAGE.MANAGER SAVING THE EVENT NOTICE
  LET HORIZON.P=PERIOD.F(TIME.V+H.MONTHS)
  LET PREVIOUS.P=PERIOD.F(TIME.V)-1
  FOR EACH BASE DO
     LET PROJECTION (HORIZON.P,BASE)=PROJECTION(PREVIOUS.P,
        BASE)
     LET ROTATION.POOL(HORIZON.P,BASE)=ROTATION.POOL(PREVIOUS.P,
        BASE)
     LOOP
  IF HORIZON.P < N.PERIOD,
     SCHEDULE THIS STORAGE.MANAGER IN   1 UNIT
     ALWAYS
END
```

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER: AFOSR-TR- 77 - 1006	2. GOVT ACCESSION NO. / 3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle): APPENDIX A: A SOURCE LISTING OF THE PROGRAM CODE FOR ISEM-P	5. TYPE OF REPORT & PERIOD COVERED: Final Report
	6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s): Charles R. Eisele Charles D. Laidlaw	8. CONTRACT OR GRANT NUMBER(s): F44620-76-C-0125
9. PERFORMING ORGANIZATION NAME AND ADDRESS: CONSAD Research Corporation 121 North Highland Avenue Pittsburgh PA 15206	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: 61102F 2313/A3
11. CONTROLLING OFFICE NAME AND ADDRESS: Air Force Office of Scientific Research (NL) Bolling AFB DC 20332	12. REPORT DATE: 27 April 1977 13. NUMBER OF PAGES: 85
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)	15. SECURITY CLASS. (of this report): Unclassified 15a. DECLASSIFICATION/ DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Integrated simulation
Modular design
Force structure planning
Training program requirements
Personnel assignment planning
Simulated personnel flow
Mission response evaluation

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The source listing of the Program Code for ISEM-P is an appendix under separate cover to the Final Report dated 27 April 1977 entitled "An Overview of the Prototype Integrated Simulation Evaluation Model of the Air Force Manpower and Personnel System."

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified